UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Escuela de Ingeniería Informática

Escaparate Interactivo de Actividades Turísticas

Proyecto Fin de Carrera

Autor: Diego Nieto Muñoz
Tutor: Alexis Quesada Arencibia
Cotutor: Enrique Ismael Mendoza Robaina
Ingeniería en Informática
17/09/2013

Proyecto fin de carrera de la Escuela de Ingeniería Informática de la Universidad de Las Palmas de Gran Canaria presentado por el alumno:

Diego Nieto Muñoz

Título del proyecto: Escaparate interactivo de actividades turísticas

Tutores:

Alexis Quesada Arencibia
Enrique Ismael Mendoza Robaina

Dedicatoria

A mis padres,
a mi hermana y
a mi novia.

Agradecimientos

A mis tutores Alexis y Enrique. A Enrique por colaborar y ayudarme en todo lo que ha podido con el robot y el proyecto. A Alexis por apostar desde el principio por el proyecto, por guiarme en cada fase y por dedicar un gran esfuerzo a que saliera adelante. Sin olvidarme de la gran oportunidad que supuso para mí entrar en el grupo AVORA. Una gran persona. Gracias.

A mis compañeros y amigos de clase por todo lo que hemos aprendido juntos, especialmente a los de siempre: Dani, Imanol, Mahy y Marcos. Sin ellos este camino no habría sido tan fácil y divertido. Espero que siempre estemos unidos y haciendo proyectos juntos.

A mis colaboradores en el proyecto: José Carlos y Cayetano, quienes también me han orientado y apoyado para que todo saliera mejor.

A mis amigos de siempre, a todos, pero especialmente Antonio. Siempre está.

A los chicos AVORA. Gracias a todos ellos he aprendido mucho y espero que sigamos haciendo robots juntos.

A todos los que están en el IUCTC, como Eduardo. Es muy enriquecedor desarrollar proyectos con todos ellos y en un lugar así.

Y como no, a toda mi familia y a mi novia. Gracias por ayudarme y apoyarme en todo momento desde siempre.

Índice

1.- Introducción. 18
2.- Estado del Arte. 19
2.1.- Sistemas de información oficiales. 19
2.2.- Sistemas de información de los hoteles. 20
2.3.- Sistemas de información de terceras empresas. 20
2.3.1.- Tripadvisor. 20
2.3.2.- Canarying. 21
2.3.3.- Wantudu. 22
2.4.- Robótica en el turismo. 22
2.4.1.- Guías turísticos. 22
3.- Objetivos. 25
4.- Metodología y recursos utilizados. 26
4.1.- Metodología de desarrollo. 26
4.2.- Recursos utilizados. 27
4.2.1.- Recursos hardware. 27
4.2.2.- Recursos software. 28
4.2.3.- Tecnologías utilizadas. 30
5.- El robot Karotz. 37
5.1.- Introducción. 37
5.2.- Análisis del robot. 38
5.2.1.- Características del robot. 38
5.2.2.- Aplicación *Karotz Controller*. 42
5.2.3.- Análisis de la API de trabajo de Karotz. 43
5.3.- Comunicación con el robot. 46
5.3.1.- Comunicación por voz. 47
5.3.2.- Comunicación y uso de RFID. 48
5.4.- KarotzVM. 50
6.- Sistema de recomendación. 51
6.1.- Introducción. 51
6.2.- Técnicas de recomendación. 52
6.2.1.- Filtros Colaborativos (CF). 52

6.3.- Sistema de recomendación implementado. 56
6.3.1.- Recolección de datos. 56
6.3.2.- Proceso de recomendación. 58
7.- El servidor. 62
7.1.- Especificación de requisitos de usuario. 62
7.1.1.- Descripción. 62
7.1.2.- Glosario de conceptos. 62
7.1.3.- Modelo del dominio. 63
7.2.- Especificación de requisitos software. 64
7.2.1.- Descripción. 64
7.2.2.- Glosario de conceptos. 64
7.2.3.- Modelo del diseño. 66
7.2.4.- Actores del software. 67
7.2.5.- Listados de actores y sus roles. 68
7.2.6.- Listados de actores y sus objetivos. 69
7.2.7.- Diagramas UML de casos de uso. 71
7.2.8.- Listado de casos de uso. 76
7.2.9.- Diagramas de secuencia. 78
7.2.10.- Prototipos de las interfaces de usuario. 82
7.2.11.- Base de datos. 91
7.2.12.- Modelo de despliegue. 96
7.2.13.- Diseño arquitectónico. 98
7.2.14.- Detalles de la implementación. 106
8.- Módulo de Karotz como interfaz cliente. 131
8.1.- Estructura de las aplicaciones en Karotz. 131
8.1.1.- descriptor.xml. 131
8.1.2.- screen.xml. 132
8.1.3.- util.js. 133
8.1.4.- main.js. 133
8.2.- Módulo cliente para Karotz 134
8.2.1.- Funciones 135
8.2.2.- Flujo de opciones del módulo cliente. 137
9.- Android. 138
9.1.- Introducción. 138

9.2.- Estructura de las aplicaciones en Android. 138
9.2.1.- Fundamentos. 138
9.2.2.- Componentes de aplicación. 140
9.3.- Desarrollo del módulo cliente del sistema de recomendación de actividades..... 140
9.3.1.- Introducción. 140
9.3.2.- Prototipos de las interfaces de usuario..... 141
9.3.3.- Diseño de la arquitectura. 142
10.- Resultados y conclusiones. 145
10.1.- El robot Karotz. Pros y contras. 145
10.2.- Uso de Zend Framework para el desarrollo del servidor. 145
10.3.- Desarrollo de una aplicación con sistema de usuarios. 146
10.4.- Sistema de recomendación de actividades. 147
10.5.- Desarrollo de aplicaciones en Android..... 147
10.6.- Uso de componentes, tecnologías y estrategias. 147
10.7.- Resultado final del proyecto. 148
11.- Trabajo futuro. 149
11.1.- Ampliación de los servicios del sistema. 149
11.1.1.- Servicios meteorológicos. 149
11.1.2.- Servicios de información de vuelos. 149
11.1.3.- Reporte de estadísticas más completo. 149
11.1.4.- Capacidad para realizar reservas de actividades directamente desde la web. 149
11.1.5.- Geoposicionamiento y uso del servicio de rutas de Google Maps. 150
11.1.6.- Interfaz multi-idioma..... 150
11.2.- Búsqueda de alternativas o complementarias a Karotz..... 150
11.3.- Mejora del sistema de recomendación. 150
11.4.- Desarrollo de una aplicación cliente para iOS..... 151
12.- Bibliografía. 152
13.- Anexos. 155
Anexo A. Detalle de casos de uso..... 155
Anexo B. Manual de usuario del portal web. 196
Rol de invitado..... 196
Rol de usuario..... 201
Rol de usuario registrado 207

Rol de cliente 208
Rol de hotel 215
Rol de hotel administrador 222
Rol de empresa 224
Rol de empresa administrador 225
Rol de administrador local 227
Rol tipo empresa 230
Rol de administrador local 235
Anexo C. Instalación y configuración de Karotz. 237
Anexo D. Manual de usuario para Karotz. 238
Anexo E. Manual de instalación de la aplicación para Android. 239
Anexo F. Manual de usuario de la aplicación para Android. 242
Inicio de sesión y registro 242
Panel principal y pestaña de búsqueda 243
Pestaña For you 246
Pestaña Favs 248
Pestaña Hotels 249
Detalle de una actividad 254
Localización de una actividad 257

Índice de figuras

Figura. 4.1. Metodología de desarrollo 26
Figura. 4.2. Ejemplo de código en HTML 31
Figura. 4.3. Ejemplo de código en PHP 32
Figura. 4.4. Ejemplo de código en CSS 33
Figura. 4.5. Ejemplo de código en Java 34
Figura. 4.6. Ejmplo código Javascript 36
Figura. 5.1. Flujo de puesta en marcha de Karotz 39
Figura. 5.2. Arquitectura de componentes en Karotz 41
Figura. 5.3. Ejemplo de código en Karotz haciendo TTS 47
Figura. 5.4. Ejemplo de gramática ABNF 47
Figura. 5.5. Ejemplo de código en Karotz haciendo ASR 48
Figura. 5.6. Ejemplo de código en Karotz utilizando RFID 50
Figura. 6.1. Esquema de recomendación 51
Figura. 6.2. Ejemplo de votaciones de actividades 53
Figura. 6.3. Distancia Euclídea 53
Figura. 6.4. Ejemplo de Distancia Euclídea 54
Figura. 6.5. Distancia de Minkowski 54
Figura. 6.6. Distancia de Mahalanobis 54
Figura. 6.7. Ejemplo de distancia de Mahalanobis 54
Figura. 6.8. Similitud basada en el coseno 55
Figura. 6.9. Correlación de Pearson 55
Figura. 6.10. Ejemplo de Correlación de Pearson 55
Figura. 6.11. Puntuación de un ítem basado en similitud 56
Figura. 6.12. Ítem a recomendar 56
Figura. 6.13. Conjunto de puntuaciones 56
Figura. 6.14. Espacio de valoraciones 57
Figura. 6.15. Conjunto de rate 57
Figura. 6.16. Conjunto de categorías 58
Figura. 6.17. Conjunto de preferencias del usuario 58
Figura. 6.18. Conjunto de actividades 58
Figura. 6.19. Conjunto de actividades aptas para el usuario 59
Figura. 6.20. Conjunto de usuarios 59
Figura. 6.21. Conjunto de usuarios acordes 59
Figura. 6.22. Conjuntos de ratings de usuarios 59
Figura. 6.23. Conjuntos de ratings del usuario final 59
Figura. 6.24. Conjunto de ratings a comparar del usuario final 60
Figura. 6.25. Conjunto de ratings a comparar del otro usuario 60
Figura. 6.26. Cálculo de la Distancia Euclídea normalizada en PHP 60
Figura. 6.27. Cálculo del ranking de actividades en PHP 61
Figura. 7.1. Modelo de dominio 63
Figura. 7.2. Modelo de diseño 66
Figura. 7.3. Actores del software 67
Figura. 7.4. Caso de uso del usuario 71

Figura. 7.5. Caso de uso del usuario registrado....72
Figura. 7.6. Caso de uso del usuario tipo empresa....72
Figura. 7.7. Caso de uso del usuario hotel....73
Figura. 7.8. Caso de uso usuario administrador local....73
Figura. 7.9. Caso de uso usuario administrador....74
Figura. 7.10. Caso de uso usuario anónimo....74
Figura. 7.11. Caso de uso del usuario cliente....75
Figura. 7.12. Diagrama de secuencia de visualización de actividad....78
Figura. 7.13. Diagrama de secuencia de modificar datos de usuario....79
Figura. 7.14. Diagrama de añadir actividad....80
Figura. 7.15. Diagrama de secuencia de unirse a un hotel....81
Figura. 7.16. Prototipo de interfaz principal del usuario anónimo....82
Figura. 7.17. Prototipo de interfaz principal del usuario cliente....83
Figura. 7.18. Prototipo de interfaz del menú de usuario cliente....84
Figura. 7.19. Prototipo de interfaz de actividades del usuario empresa....85
Figura. 7.20. Prototipo de interfaz principal del usuario hotel....86
Figura. 7.21. Prototipo de interfaz principal del usuario cliente buscando o listando actividades....87
Figura. 7.22. Prototipo de interfaz para el usuario hotel listando o buscando actividades....88
Figura. 7.23. Prototipo de interfaz de usuario del usuario hotel listando notificaciones....89
Figura. 7.24. Prototipo de interfaz de usuario hotel creando una nueva notificación....90
Figura. 7.25. Esquema de la base de datos....91
Figura. 7.26. Base de datos. Interrelación activity-filtered-company....93
Figura. 7.27. Base de datos. Interrelación activity-favorite-user....94
Figura. 7.28. Base de datos. Interrelación user-company....95
Figura. 7.29. Modelo de despliegue....96
Figura. 7.30. Arquitectura de la solución....98
Figura. 7.31. Arquitectura servidor....99
Figura. 7.32. Módulos de Zend Framework....101
Figura. 7.33. Arquitectura modelo-vista-controlador....102
Figura. 7.34. Flujo de ejecución en Zend Framework....105
Figura. 7.35. Estructura de la aplicación. Nivel cero....108
Figura. 7.36. Estructura de la aplicación. Application....109
Figura. 7.37. Estructura de la aplicación. Public....110
Figura. 7.38. Estructura de la aplicación. Tests....111
Figura. 7.39. Campos y contenido de los usuarios en la base de datos....113
Figura. 7.40. Proceso de registro....114
Figura. 7.41. Código de registro en el sistema....114
Figura. 7.42. Proceso de login en el sistema....115
Figura. 7.43. Código de login en el sistema....116
Figura. 7.44. Código de logout....116
Figura. 7.45. Esquema de recursos del sistema....117
Figura. 7.46. Código de algunos recursos de la aplicación....117

Figura. 7.47. Esquema de roles en Zend Framework 118
Figura. 7.48. Código definiendo los roles en Zend Framework 118
Figura. 7.49. Código dando los permisos en Zend Framework 119
Figura. 7.50. Ejemplo CSRF parte 1 122
Figura. 7.51. Ejemplo CSRF parte 2 122
Figura. 7.52. Ejemplo CSRF parte 3 122
Figura. 7.53. Flujo de Zend Validation Hash 122
Figura. 7.54. Código para el tratamiento de imágenes en el sistema 124
Figura. 7.55. Ejemplo de código para AJAX en Zend Framework 125
Figura. 7.56. Ejemplo de PHP en Zend Framework 126
Figura. 7.57. Código para testear el proceso de login con PHPUnit 128
8.1. Ejemplo de descriptor de fichero de Karotz 132
Figura. 8.2. Descriptor de los paneles de configuración de la aplicación para Karotz 133
Figura. 8.3. Fichero de conexión a Karotz 133
Figura. 8.4. Ejemplo de código principal para Karotz 134
Figura. 8.5. Flujo de opciones en la aplicación para Karotz 137
Figura. 9.1. Arquitectura de Android 138
Figura. 9.2. Prototipo de interfaz en Android 141
Figura. 9.3. Transferencia de datos entre el cliente y el servidor 143

Índice de tablas

Tabla 6.1. Ejemplo de scores de actividades 57
Tabla 6.2. Ejemplo de ratings de actividades 57
Tabla 6.3. Ejemplo de preferencias de usuarios 58
Tabla 7.1. Listado de actores y sus roles 68
Tabla 7.2. Listados de actores y sus objetivos 71
Tabla 7.3. Listado de casos de uso 77
Tabla 7.4. Componentes de Zend-Framework utilizados 107
Tabla 7.5. Funciones de Google Maps 121
Tabla 7.6. Métodos de Google Maps empleados 121
Tabla 7.7. Métodos para trabajar con AJAX en Zend Framework 125
Tabla 7.8. Parámetros de la petición AJAX con JQuery 126
Tabla 7.9. Métodos de Google Charts utilizados 130
Tabla 9.1. Actividades en Android 142
Tabla 9.2. Clases de utilidad en Android 143

Índice de imágenes

Imagen 2.1. Tríptico de actividades oficiales 19
Imagen 2.2. Portal de turismo de Gran Canria 19
Imagen 2.4. Portal de actividades de Tripadvisor 21
Imagen 2.3. Tabla de actividades de un hotel 20
Imagen 2.5. Portal web de Canarying 21
Imagen 2.6. Pantalla táctil de la empresa Wantudu 22
Imagen 2.8. Robot EMIEW 2 23
Imagen 2.7. Robot RoboX 23
Imagen 2.9. Robots del banco Santander Interactive Guest Assistants 24
Imagen 4.1. Escritorio Microsoft Windows 7 28
Imagen 4.2. Escritorio Ubuntu 12.04 28
Imagen 4.3. Aplicación Oracle MySQL Workbench 29
Imagen 4.4. IDE Netbeans 29
Imagen 4.5. IDE Eclipse 29
Imagen 4.6. IDE Microsoft Visual Studio 30
Imagen 5.1. Robot Karotz de frente 37
Imagen 5.2. Partes frontales del Robot Karotz 38
Imagen 5.3. Partes traseras del robot Karotz 39
Imagen 5.4. Aplicación Karotz Controller 42
Imagen 5.5. Conjunto de Flatnanoz 49
Imagen 5.6. Conjunto de Nanoztag 49
Imagen 7.1. Captura Google Maps IUCTC 119
Imagen 7.2. Captura de pantalla de estadísticas de un hotel 129
Imagen 7.3. Captura de pantalla de estadísticas de una actividad 130
Imagen 13.1. Interfaz de rol invitado 196
Imagen 13.2. Registro de un usuario cliente 197
Imagen 13.3. Mensaje de registro correcto 197
Imagen 13.4. Mensaje de fallo en el registro y formulario 198
Imagen 13.5. Registro de una compañía 199
Imagen 13.6. Formulario de acceso 200
Imagen 13.7. Formulario de acceso fallido 200
Imagen 13.8. Detalle de actividad para el rol anónimo 201
Imagen 13.9. Listado de actividades 202
Imagen 13.10. Interfaz de *Top 10 activities* 203
Imagen 13.11. Módulo de búsqueda. Campo de texto 204
Imagen 13.12. Módulo de búsqueda. Categoría 204
Imagen 13.13. Módulo de búsqueda. Datepicker 205
Imagen 13.14. Módulo de búsqueda. Timepicker 205
Imagen 13.15. Módulo de búsqueda. Tooltip precio 205
Imagen 13.16. Módulo de búsqueda. Resultados y paginación 206
Imagen 13.17. Spinner 206
Imagen 13.18. Lista actividades. Mostrar más 207
Imagen 13.19. Ver o modificar el perfil del usuario 207

Imagen 13.20. Modificar contraseña del usuario 208
Imagen 13.21. Panel principal del usuario cliente 208
Imagen 13.22. Detalle de una actividad de un usuario cliente 210
Imagen 13.23. Añadir actividad a favoritos 211
Imagen 13.24. Eliminar actividad de favoritos 211
Imagen 13.25. Votar una actividad 211
Imagen 13.26. Votar una actividad. Número de estrellas 212
Imagen 13.27. Acciones del hotel del cliente sin notificaciones 212
Imagen 13.28. Acciones del hotel del cliente con notificaciones 213
Imagen 13.29. Listado de notificaciones 213
Imagen 13.30. Botón *Hotels* de la barra superior de navegación 214
Imagen 13.31. Búsqueda y unión a hoteles 214
Imagen 13.32. Menú de usuario cliente 215
Imagen 13.33. Ver y modificar preferencias 215
Imagen 13.34. Panel principal del usuario hotel 216
Imagen 13.35. Menú de acciones del usuario hotel 216
Imagen 13.36. Crear nueva notificación 217
Imagen 13.37. Crear nueva notificación. Mensaje finalización 217
Imagen 13.38. Lista de notificaciones publicadas por el hotel 218
Imagen 13.39. Eliminación de notificación 218
Imagen 13.40. Modificación de una notificación 219
Imagen 13.41. Modificación de una notificación. Mensaje de finalización 219
Imagen 13.42. Añadir actividad a la lista de sugeridas 220
Imagen 13.43. Listado de actividades sugeridas 220
Imagen 13.44. Eliminar actividad sugerida 221
Imagen 13.45. Panel principal del usuario hotel administrador 222
Imagen 13.46. Menú de acciones del usuario hotel administrador 222
Imagen 13.47. Estadísticas del hotel 223
Imagen 13.48. Estadísticas de una actividad del hotel 224
Imagen 13.49. Panel principal del usuario empresa 224
Imagen 13.50. Menú de acciones del usuario empresa 225
Imagen 13.51. Panel principal del usuario empresa 225
Imagen 13.52. Menú de acciones del usuario empresa administrador 226
Imagen 13.53. Estadísticas del usuario empresa administrador 226
Imagen 13.54. Estadística de una actividad 227
Imagen 13.55. Crear nuevo usuario de hotel 227
Imagen 13.56. Crear nuevo usuario empresa 228
Imagen 13.57. Crear un nuevo usuario de la compañía. Mensaje de confirmación 228
Imagen 13.58. Lista de usuarios de un hotel 228
Imagen 13.59. Lista de usuarios de una empresa 229
Imagen 13.60. Actualización de un usuario de la compañía 229
Imagen 13.61. Actualización de un usuario de la compañía. Mensaje de confirmación 229
Imagen 13.62. Eliminación de un usuario 230
Imagen 13.63. Crear actividad (arriba) 231

Imagen 13.64. Crear actividad (abajo) 232
Imagen 13.65. Crear actividad. Mensaje de confirmación 232
Imagen 13.66. Lista de actividades de un hotel 233
Imagen 13.67. Lista de actividades de una empresa 233
Imagen 13.68. Modificar una actividad 234
Imagen 13.69. Modificar una actividad. Mensaje de confirmación 234
Imagen 13.70. Eliminar una actividad 235
Imagen 13.71. Menú de usuario de una empresa y un hotel 235
Imagen 13.72. Actualizar perfil de la compañía 236
Imagen 13.73. Actualizar perfil de la compañía. Mensaje de error 236
Imagen 13.74. Actualizar perfil de la compañía. Mensaje de confirmación 236
Imagen 13.75. Encendido de Karotz 238
Imagen 13.76. Iniciar la aplicación en Karotz 238
Imagen 13.77. Búsqueda de EIAT en Google Play 239
Imagen 13.78. Página de EIAT en Google Play 240
Imagen 13.79. Instalación de EIAT en el dispositivo Android 241
Imagen 13.80. Android. Pantalla de login 242
Imagen 13.81. Android. Pestaña search 243
Imagen 13.82. Android. Menú contextual 244
Imagen 13.83. Android. Cuadros de diálogo 244
Imagen 13.84. Android. Pestaña search. Comienzo 245
Imagen 13.85. Android. Pestaña For you 246
Imagen 13.86. Android. Pestaña For you. Top ten activities 247
Imagen 13.87. Android. Pestaña Favs 248
Imagen 13.88. Android. Pestaña hotels. Búsqueda de hotel 249
Imagen 13.89. Android. Pestaña Hotels. Lista de hoteles 250
Imagen 13.90. Android. Mensaje de unión a hotel 250
Imagen 13.91. Android. Pestaña Hotels. Menú de hotel 251
Imagen 13.92. Android. Pestaña hotels. Notificación 251
Imagen 13.93. Android. Pestaña hotels. Lista de notificaciones 252
Imagen 13.94. Android. Mensaje de desunión del hotel 253
Imagen 13.95. Android. Detalle de actividad (arriba) 254
Imagen 13.96. Android. Detalle de actividad (abajo) 255
Imagen 13.97. Android. Detalle de actividad. Botón quitar de favoritos 256
Imagen 13.98. Android. Detalle de actividad. Votación 256
Imagen 13.99. Android. Localización de actividad 257

1.- Introducción.

En este proyecto se ha pretendido desarrollar un completo sistema de gestión de actividades turísticas. Para ello se ha investigado cómo los hoteles y las empresas interactúan con los turistas y les muestran su oferta de ocio.

Como resultado de esa investigación se propone un sistema de recomendación de actividades a los turistas. Este sistema contiene las actividades de las empresas de ocio y hoteles que se registran. La aplicación ha sido diseñada de forma que los usuarios de los diferentes roles puedan acceder y realizar sus funciones mediante un sistema de autenticación y autorización de usuarios.

El sistema utiliza un algoritmo de *Collaborative Filtering* similar al de Amazon para recomendar las actividades a los usuarios. Estos usuarios pueden acceder a las actividades y recomendaciones del sistema mediante tres vías: un portal web, un cliente para Android y un cliente para el robot Karotz.

En la aplicación los hoteles pueden tener usuarios unidos a su hotel para poder enviarles notificaciones y mostrar sus actividades.

Todas las compañías, empresas y hoteles, gestionan sus actividades y pueden ver sus estadísticas.

2.- Estado del Arte.

A continuación se van a mostrar las diferentes vías de información de actividades que existen la actualidad. Para ello nos centraremos en tres categorías: sistemas de información de actividades oficiales, sistemas de información de hoteles y sistemas de información de terceras empresas. Finalmente se mostrarán algunos ejemplos de robots aplicados en el turismo.

2.1.-Sistemas de información oficiales.

En la actualidad los sistemas oficiales de información en Gran Canaria tratan de informar al turismo por dos formas diferentes: vía web y por pequeños documentos o trípticos.

Imagen 2.1. Tríptico de actividades oficiales

En los puntos de información turística de la isla se proporcionan mapas y carteleras de eventos. Estos puntos están distribuidos por toda la capital para facilitar al viajero su disponibilidad y cercanía. La mayoría de los eventos que ofrecen pertenecen a actividades oficiales y no a terceras empresas, como por ejemplo ferias, convenciones y actividades culturales.

Por otra parte, sí existe un directorio clasificado de actividades en el portal web del Patronato de Turismo de Gran Canaria. En él se proporciona un listado de empresas con las que se puede contactar para realizar las actividades. Además se ofrecen otro tipo de recursos como mapas y guías para el turista. El sitio está ofrecido en más de ocho idiomas, entre los que se encuentran los más populares para los turistas.

Imagen 2.2. Portal de turismo de Gran Canria

2.2.-Sistemas de información de los hoteles.

Los hoteles de menor categoría, por lo general, continúan ofreciendo sus actividades mediante trípticos o documentos a la llegada al hotel. La información contenida suele ser muy escueta y comprimida para ahorrar costes. Además, es habitual que en la recepción del hotel se mantenga una vitrina para la publicación de estas actividades.

Los hoteles de mayor categoría, además de las vías anteriores, tienen incorporado en sus portales web un apartado donde muestran de qué actividades dispone el hotel. En la descripción de esas actividades suele estar detallado tanto los horarios como los precios. Las reservas, como canchas de pádel o tenis, se realizan en el mismo hotel con un tiempo de antelación.

HORAS	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SABADO	DOMINGO
10:30 A 11:00	AQUAGYM	AQUAGYM	AQUAGYM	AQUAGYM	AQUAGYM	AQUAGYM	AQUAGYM
11:00 A 12:00	TALLER DE DIBUJO	TALLER DE BOLAS MALABARES	TALLER DE ARCILLA	TALLER DE CARIOCAS	TALLER DE MASCARAS	TALLER DE MARCOS DE FOTO	TALLER DE BISUTERIA
12:00 A 13:00	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS
DESCANSO	DE	13:00	A	15:30	PARA	COMER	-
15:30 A 16:30	TALLERES	TALLERES	TALLERES	TALLERES	TALLERES	TALLERES	TALLERES
16:30 A 17:30	ACTIVIDADES DEPORTIVAS	ACTIVIDADES DEPORTIVAS	ACTIVIDADES DEPORTIVAS	ACTIVIDADES DEPORTIVAS	ACTIVIDADES DEPORTIVAS	ACTIVIDADES DEPORTIVAS	ACTIVIDADES DEPORTIVAS
17:30 A 18:30	PISCINA	PISCINA	PISCINA	PISCINA	PISCINA	PISCINA	PISCINA
18:30 A 19:00	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS	JUEGOS VARIOS

Imagen 2.3. Tabla de actividades de un hotel

2.3.-Sistemas de información de terceras empresas.

Los sistemas de información de terceras empresas son los que más han evolucionado y proliferado recientemente. Las empresas más populares en Gran Canaria y relacionadas con este proyecto se comentan a continuación.

2.3.1.- Tripadvisor

La web de viajes más grandes del mundo ya no sólo se dedica a mostrar las opiniones de los usuarios en los hoteles, ahora también tiene un directorio de actividades de los lugares. Los usuarios y negocios pueden crear actividades de los sitios y poner su opinión. Estas actividades son localizadas geográficamente y revisadas por los gestores de Tripadvisor para validar su autenticidad.
Las actividades creadas son comentadas y valoradas por los usuarios. Estos además pueden añadir sus fotos referentes a la actividad.

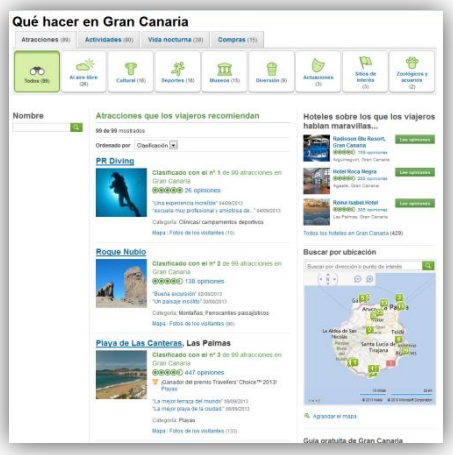


Imagen 2.4. Portal de actividades de Tripadvisor

2.3.2.- Canarying

Canarying es un nuevo portal web centrado en Gran Canaria que pone a disposición del usuario todo un directorio de actividades. Estas actividades pertenecen a terceras empresas que actúan como proveedores del sistema. Casi todas las actividades son reservables y ofrecen información detallada como la localización, su disponibilidad y una descripción.

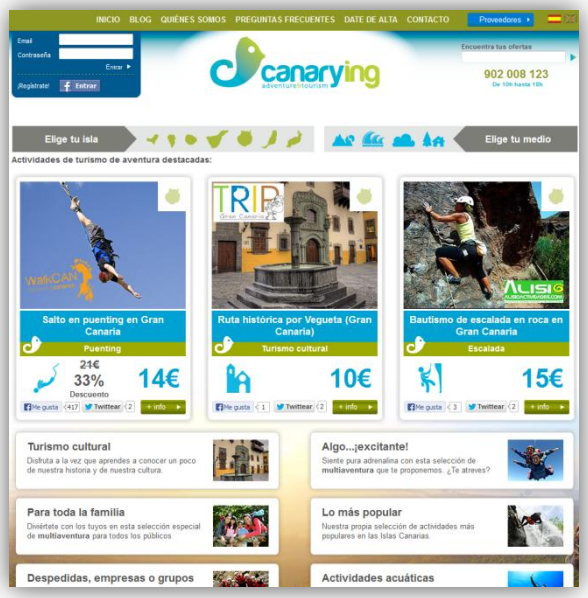


Imagen 2.5. Portal web de Canarying

2.3.3.- Wantudu

La empresa Wantudu es pionera en Gran Canaria en ofrecer un servicio de actividades de la isla. Es una solución para hoteles con el fin de la fidelización de clientes a través de ofrecerles información de ocio. Sus servicios comprenden la instalación de una pantalla táctil en la recepción del hotel y una aplicación para móvil. La información que proporciona ocupa desde la meteorología en la zona hasta una completa gama de actividades organizadas por temática.

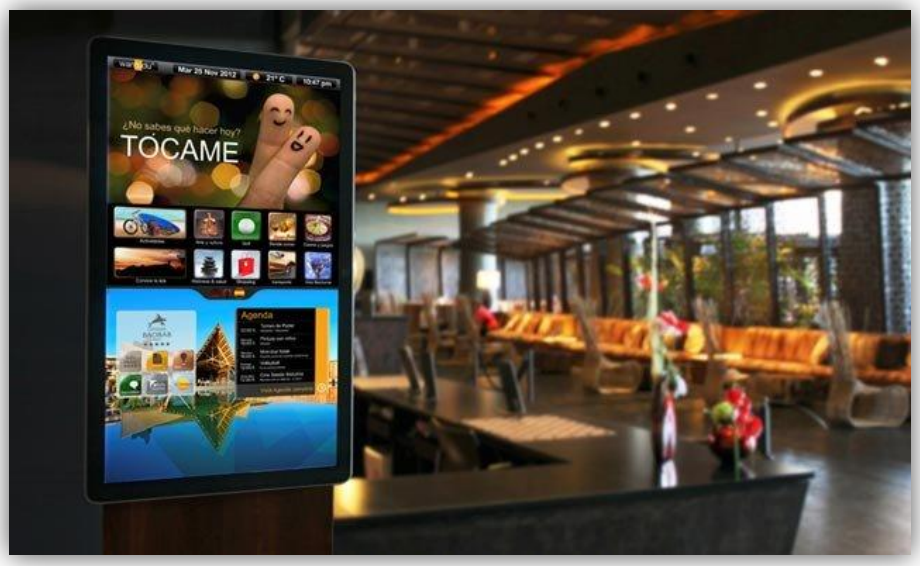


Imagen 2.6. Pantalla táctil de la empresa Wantudu

Existen otros portales como Gran Canaria Info y La Guía de Gran Canaria que también actúan como un directorio de actividades de la isla.

2.4.-Robótica en el turismo.

La robótica en el turismo no está demasiado desarrollada aunque hay algunos ejemplos de desarrollos realizados. La podemos dividir en guías turísticos, que son lo que están en contacto casi permanente con el turista, y los asistentes, que son aquellos que ayudan en tareas más concretas al turista.

2.4.1.- Guías turísticos.

Los robots que tienen la función de hacer de guías turísticos mantienen un contacto muy estrecho con el turista. Pueden ser guías en los hoteles, museos y centros comerciales, recepcionista o cualquier otro tipo de función que tenga como objetivo orientar al turista.

Este tipo de robots facilitan la vida del turista ya que guían o dan la información que necesitan en cualquier momento y de forma rápida. Se muestran a continuación algunos ejemplos.

RoboX

Imagen 2.7. Robot RoboX

RoboX es un robot móvil interactivo capaz de relacionarse con humanos. Su principal función es servir como asistente turístico en hoteles, museos o ferias, guiando a los visitantes por sus instalaciones y respondiendo a sus preguntas.

Es totalmente autónomo y es capaz de hablar en 4 idiomas diferentes. Puede ejercer sus funciones durante 12 horas al día durante los 7 días de la semana. También muestra sus "emociones" mediante iconos en sus ojos y pequeñas animaciones.

La gran ventaja de este robot es su capacidad de navegación en lugares con muchas personas como ferias o exhibiciones ya que es capaz de manejar 5000 visitantes al día. Debido a esto, podría servir también para realizar campañas de marketing y publicitar productos.

EMIEW 2

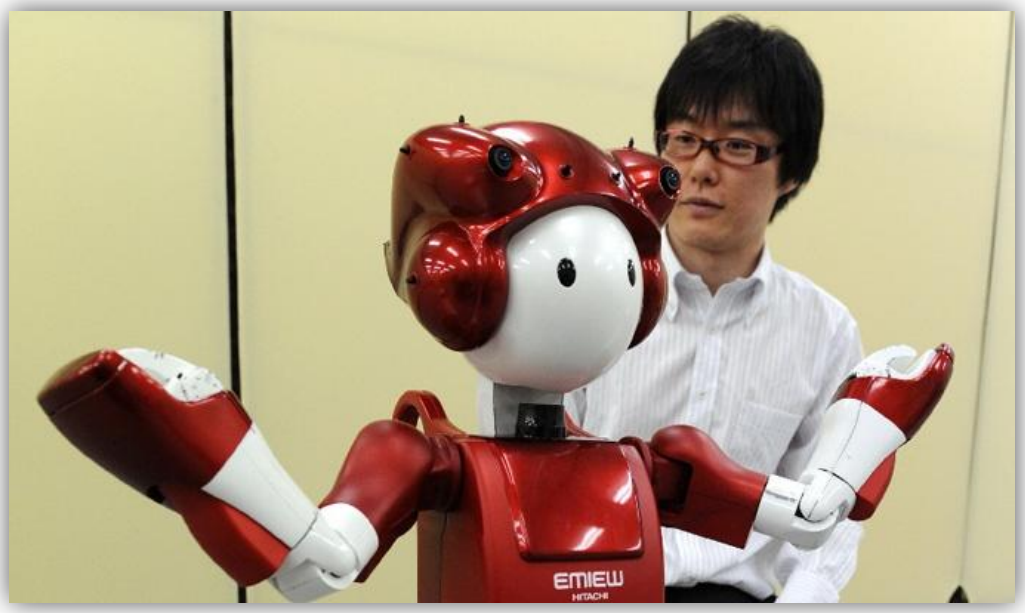


Imagen 2.8. Robot EMIEW 2

EMIEW 2 es la nueva versión de EWIEW. Está pensado para trabajar dentro de un ambiente de oficina midiendo 80cm de alto y pesando 14K. Es un robot diseñado para ayudar a las personas en su vida diaria y actuar mediante órdenes de humanos. Algunos de los servicios que aporta son guía, acompañante o ayudante.

Las habilidades que posee este robot son:

- Gran velocidad y agilidad en sus movimientos
- Reconocimiento de objetos
- Reconocimiento de voz has en ambientes ruidosos.
- Brazo con movilidad humana

Interactive Guest Assistants

Imagen 2.9. Robots del banco Santander Interactive Guest Assistants

Así es como se le conoce a toda una flota de robots que ha comprado el banco Santander a la empresa YDreams. Estos robots son capaces de guiar a los visitantes a la parte deseada del banco. Se controlan por Wi-Fi y son operados por una pantalla táctil de manera que permiten al usuario elegir su idioma o acceder a datos de audio y video de la historia del banco o del grupo Santander.

3.- Objetivos.

Actualmente el turismo es uno de los grandes motores de la economía española, y más en las Islas Canarias. Por ello, es importante tener siempre una cara moderna, personalizada y atractiva para nuestros visitantes. El punto final de este proyecto es la creación de un sistema que permita a los usuarios de los hoteles obtener información acerca de actividades y eventos en los propios hoteles y empresas cercanas al hotel. Para ello es fundamental conocer los gustos del cliente y ofrecer en función de los mismos los servicios más adecuados a su perfil. Los medios de difusión serán tres clientes: la plataforma robótica Karotz, a partir de la cual es posible interactuar con ella gracias a su conectividad a internet; un portal web; y una aplicación móvil para dispositivos Android, con la que será posible acceder a los servicios de información de actividades y situarlas en un mapa. Para que todo esto sea posible es necesario abordar los siguientes subobjetivos:

- Servidor para alojar, registrar y actualizar toda la información referente a los hoteles, usuarios y empresas. Así como la interfaces que harán de puente entre la base de datos y los dispositivos clientes.
- Desarrollo de la interfaz de reconocimiento de voz de los usuarios de forma generalista.
- Desarrollo de las aplicaciones tanto en la plataforma robótica Karotz como en la aplicación móvil correspondientes a:
    - Informar de actividades externas según la categoría seleccionada, en Karotz mediante la interacción oral con el robot.
    - Indicar actividades, eventos u otras circunstancias que se pudieran dar mediante la visualización o escucha de actividades y notificaciones del hotel.
- Mostrar el mapa de actividades en el cliente web y Android.
- Desarrollo de una interfaz web para facilitar a las empresas, usuarios y hoteles registrarse y gestionar sus datos.

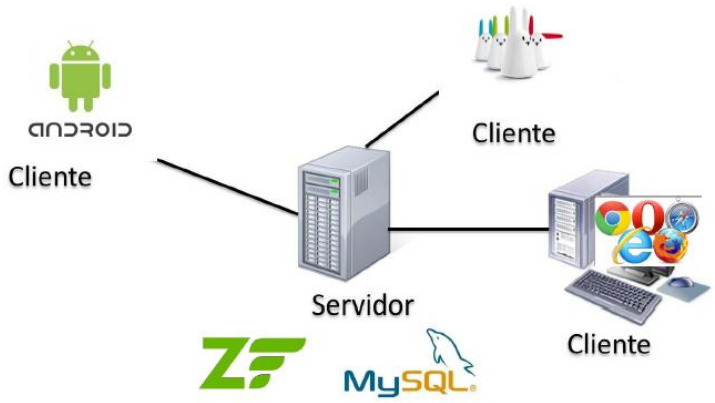


Figura 3.1. Arquitectura de la solución

4.- Metodología y recursos utilizados.

4.1.-Metodología de desarrollo.

En este proyecto se ha optado por seguir un desarrollo de prototipos combinándolo con un desarrollo en cascada. Por tanto se han seguido las siguientes fases:

1. Desarrollo de prototipos.
2. Análisis de requisitos de usuario.
   Definición de lo que el usuario espera del sistema. A través de entrevistas, encuestas estudios y prototipados.
3. Análisis de requisitos software.
   En este apartado se trata de obtener una definición del software que se va a construir. Se construye un modelo lógico del software.
4. Diseño arquitectónico.
   Se define la estructura del software, equivale a la estructura más su control. Se especifica cada módulo y se diseñan las interrelaciones entre ellos.
5. Diseño detallado y producción.
   Se detalla cada componente de la arquitectura definido anteriormente. A continuación, se produce en los lenguajes de programación elegidos el sistema software para su posterior integración y testeo.
6. Transferencia.
   Se implanta el sistema y se configura en un entorno operativo para dejarlo ejecutable para el usuario.

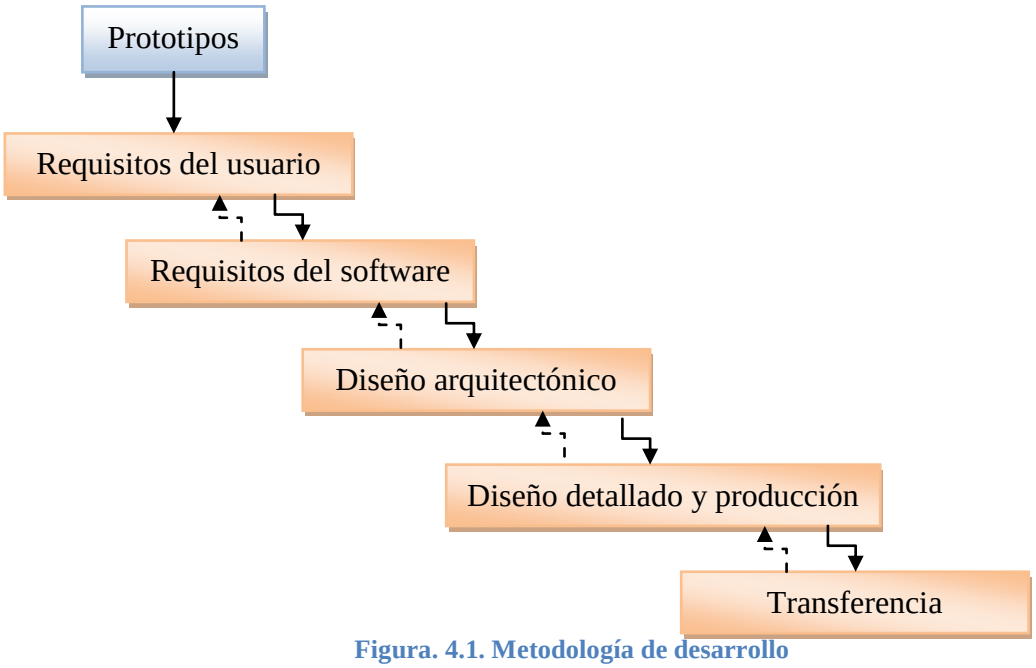


Figura. 4.1. Metodología de desarrollo

El desarrollo del prototipo ha seguido las siguientes fases:

1. Recolección de requisitos iniciales.
2. Diseño rápido.
3. Construcción del prototipo.
4. Evaluación del prototipo.

Esta primera metodología de desarrollo aplicada ha permitido identificar claramente los requisitos del usuario y evaluar los límites que podía alcanzar el proyecto.

La segunda metodología usada, la clásica o en cascada, es el método más asentado por su historia y ha sido fuente de inspiración para otros modelos de proceso de software. Este modelo exige un enfoque sistemático y secuencial. En el que cada fase se ejecuta una vez y sienta las bases para la siguiente. Esto no quiere decir que en determinadas fases no se haga una realimentación a las anteriores, de hecho a medida que se va completando cada fase se van ajustando las anteriores.

4.2.-Recursos utilizados.

4.2.1.- Recursos hardware.

Equipo informático:

- Procesador: Intel® Core™ i7 CPU 860 @ 2,8Ghz
- Memoria RAM: 4,00GB
- Disco duro: 1TB
- Tarjeta gráfica: GeForce 7300 SE
- Pantalla panorámica de 22’’
- Teclado y ratón

Robots Karotz:

- Dos unidades de robots Karotz
  - Procesador: ARM9 Processor de 400Mhz.
  - Memoria: 64MB de memoria RAM. 256MB de memoria Flash.
  - Micrófono
  - Altavoz
  - Webcam
  - 2 Servomotores

4.2.2.- Recursos software.

SO - Microsoft Windows 7

Windows 7 es una versión de Microsoft Windows, línea de sistemas operativos producida por Microsoft Corporation. Esta versión está diseñada para uso en PC, incluyendo equipos de escritorio en hogares y oficinas, equipos portátiles, tablet PC, netbooks y equipos media center. Incluye varias características nuevas, como mejoras en el reconocimiento de escritura a mano, soporte para discos duros virtuales, rendimiento mejorado en procesadores multinúcleo, mejor rendimiento de arranque, DirectAccess y mejoras en el núcleo.

Imagen 4.1. Escritorio Microsoft Windows 7

El desarrollo de las aplicaciones del sistema se ha llevado a cabo mayoritariamente en este sistema operativo.

SO - Ubuntu Linux.

Imagen 4.2. Escritorio Ubuntu 12.04

Ubuntu es un sistema operativo basado en Debian y que se distribuye como software libre y gratuito, el cual incluye su propio entorno de escritorio denominado Unity. Está orientado al usuario novel y promedio, con un fuerte enfoque en la facilidad de uso y en mejorar la experiencia de usuario. Está compuesto de múltiple software normalmente distribuido bajo una licencia libre o de código abierto. Ha formado parte de los SO en los que se ha desarrollado el proyecto.

Oracle MySQL Server

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. En este caso como la aplicación web estaba desarrollada en PHP se ha optado también por utilizar este servidor de bases de datos.

Oracle MySQL Workbench

MySQL Workbench es una herramienta visual de diseño de bases de datos que integra desarrollo de software, Administración de bases de datos, diseño de bases de datos, creación y mantenimiento para el sistema de base de datos MySQL. Es el sucesor de DBDesigner 4 de fabFORCE.net, y reemplaza el anterior conjunto de software, MySQL GUI Tools Bundle. Ha sido clave para diseñar y gestionar la base de datos de la aplicación web.

Imagen 4.3. Aplicación Oracle MySQL Workbench

Apache.

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.12 y la noción de sitio virtual. Es un servidor modular, de código abierto, multi-plataforma y extensible. Por todo esto fue la elección tomada para este proyecto.

Netbeans

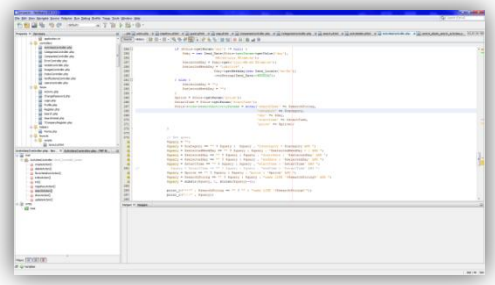


Imagen 4.4. IDE Netbeans

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo para programar en otros lenguajes como PHP, C++ y HTML. En el caso de este proyecto ha sido el entorno de desarrollo para toda fase de desarrollo de la aplicación web.

Eclipse

Eclipse es un programa informático compuesto por un conjunto de herramientas de programación de código abierto. El IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) se entrega como parte de Eclipse. En este entorno se ha desarrollado la aplicación cliente del sistema en Android implementándolo en lenguaje Java.

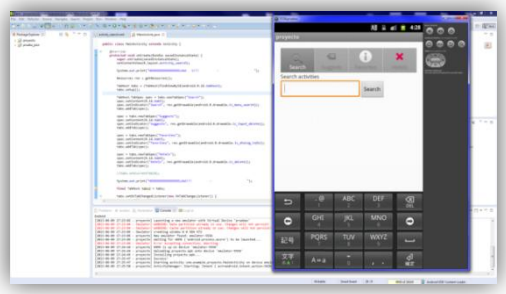


Imagen 4.5. IDE Eclipse

Microsoft Visual Studio 2010

Microsoft Visual Studio es un entorno de desarrollo para sistemas operativos Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, y Visual Basic .NET, al igual que entornos de desarrollo web como ASP.NET. aunque actualmente se han desarrollado las extensiones necesarias para muchos otros. En el caso de este proyecto sólo se ha utilizado para diseñar los diagramas de clase y casos de uso ya que tenía una herramienta potente para este cometido.

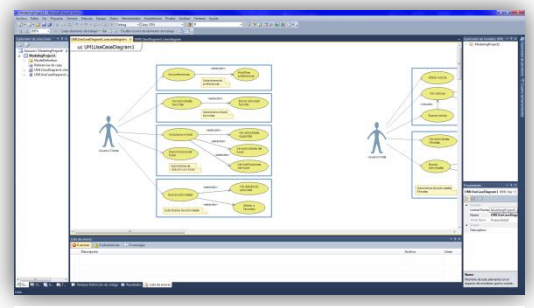


Imagen 4.6. IDE Microsoft Visual Studio

KarotzVM SDK

Es una máquina virtual para Karotz, un fichero jar. Permite testear las aplicaciones de manera simulada o en el propio robot sin tener que enviar la aplicación a la Application Store de Karotz. Muestra en la propia consola las acciones que se van tomando durante la ejecución del programa y los fallos que puedan producirse.

Android SDK

Es el software de desarrollo que te ofrece las librerías, herramientas de desarrollo, compilación y testeo que se necesitan para desarrollar aplicaciones en Android. Se divide en varios módulos:

- ADT plugin, que es una extensión para el IDE de desarrollo de Eclipse.
- Android SDK tools, con las herramientas para gestionar las versiones de desarrollo instaladas y que se podrán utilizar en Eclipse.
- Emulador de dispositivos Android, que permite la ejecución de las aplicaciones como si se tratase de un dispositivo real.

Git

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Ha sido un elemento clave en el desarrollo del proyecto ya que, sobre todo en la implementación del servidor, estaba compuesto de una gran cantidad de ficheros que se modificaban continuamente y necesitaban ser gestionados.

4.2.3.- Tecnologías utilizadas.

HTML

Representa toda la estructura en la que está fundamentada la web hoy en día. Sus siglas significan *HyperText Markup Languaje* y hace referencia al lenguaje de marcado para la elaboración de páginas web. Este lenguaje se escribe en forma de etiquetas y se encarga de definir y estructurar cada uno de los elementos que aparecen en una página web. Fue publicado por primera vez en Internet por Tim Berners-Lee en 1991.

Actualmente la W3C se encarga de desarrollar estándares abiertos para asegurar el correcto crecimiento de la web. La última versión de HTML, HTML 5, ofrece nuevas etiquetas para incrustar videos y gráficos de manera directa en la propia página entre otras muchas nuevas características.

La estructura de las páginas web tiene siempre la siguiente forma:

```
<!DOCTYPE HTML>
 <html>
   <head>
     <title>Ejemplo1</title>
   </head>
   <body>
     <p>ejemplo1</p>
   </body>
 </html>
```

Figura. 4.2. Ejemplo de código en HTML

Donde entre la cabecera <head></head> se incluyen todos los scripts y definiciones de código que luego se usarán a lo largo de la página web, que queda desarrollado en el cuerpo de la página <body></body>.

PHP

PHP es un acrónimo recursivo y significa *PHP Hypertext Pre-procesor*. Es un lenguaje de programación de uso general que trabaja del lado del servidor y fue diseñado para el desarrollo web de contenido dinámico. Es uno de los primeros lenguajes que permitió incorporar código directamente en el HTML en vez de un fichero externo que procesara datos. Forma parte de la familia de lenguajes interpretados y el servidor web es quien se encarga de procesarlo y generar la página web resultante. Es posible utilizarlo tanto en sistemas UNIX como en Microsoft Windows apoyándose en servidores como Apache, CGI e ISAPI.

Es un lenguaje que derivó en su nacimiento (1995) de los lenguajes más comunes de programación estructurada como C y Perl. Actualmente es uno de los lenguajes más utilizados en el desarrollo en la web y constituye una alternativa a otros lenguajes como Microsoft ASP y las páginas JSP de Java.

PHP tiene incorporados unos puentes que permiten realizar conexiones con diferentes tipos de bases de datos como MySQL, PostgreSQL, Oracle,ODBC, Microsoft SQL Server, Firebird y SQLite.

En la siguiente porción de código se encuentra incrustado código PHP en HTML:

```
<html>
 <head>
  <title>Ejemplo PHP</title>
 </head>
 <body>
 <?php echo '<p>Hola Mundo</p>'; ?>
 </body>
</html>
```

Figura. 4.3. Ejemplo de código en PHP

AJAX

AJAX no es en sí un lenguaje de programación sino una técnica de desarrollo web para crear aplicaciones interactivas. Como su propio nombre indica, Asynchronous JavaScript And XML, utiliza el lenguaje de programación JavaScript y la estructuración de datos XML en el lado del cliente para comunicarse de forma asíncrona con el servidor. Esta tecnología permite modificar elementos de la página web sin necesidad de recargarla o ir a otra página. De esta forma es posible incrustar datos recuperados de la B.B.D.D. en un elemento concreto de la página. Su principal ventaja es que evita la necesidad de cargar todo el contenido cuando sólo es necesario refrescar o cargar algunos elementos de la página.

Su funcionamiento está basado en peticiones realizadas mediante XMLHttpRequest y procesado y modificado de elementos en la parte del cliente con JavaScript y DOM.

CSS

Las hojas de estilo en cascada son un lenguaje que permiten describir estilos de un documento (página web) a través de marcas. Esta información puede estar incrustada en el propio HTML o por el contrario separada en ficheros .css.

Su sintaxis está basado en uno o más selectores y un bloque de estilos donde se definen los valores de las propiedades del documento. Los selectores pueden ser únicos cuando se trata del *id* de un elemento o múltiple cuando se habla de una clase de elementos.

De acuerdo con su definición no es posible usar selectores en arden ascendente en el DOM. Además no están declaradas reglas de cálculo numérico que permitan especificar valores. Pero en contra partida podemos decir que entre sus ventajas es posible encontrarse con un control centralizado de la presentación de

los sitios web, de manera que se agiliza de forma considerable la actualización del mismo. Es posible la separación del estilo del contenido, lo que habilita al desarrollador a modificar el contenido

El siguiente *snippet* de código nos muestra un selector con su bloque y las propiedades que modifica:

```
.registerLogin {
    width: 500px;
}
```

Figura. 4.4. Ejemplo de código en CSS

SQL

El SQL es un lenguaje de consulta estructurado de acceso a bases de datos relaciones que permite especificar diversos tipos de operaciones entre ellas. Una de sus características es el manejo del álgebra y el cálculo relaciones que permiten efectuar consultas con el fin de recuperar, de forma sencilla, información en bases de datos, así como hacer cambios en ella.

En SQL una sola sentencia puede equivaler a uno o más programas que se utilizarían en un lenguaje de bajo nivel orientado a registros. Es posible dividir este lenguaje en:

- **Lenguaje de definición de datos**: El LDD de SQL proporciona comandos para la definición de esquemas de relación, borrado de relaciones y modificaciones de los esquemas de relación.
- **Lenguaje interactivo de manipulación de datos**: El LMD de SQL incluye lenguajes de consultas basado tanto en álgebra relacional como en cálculo relacional de tuplas.
- **Integridad**: El LDD de SQL incluye comandos para especificar las restricciones de integridad que deben cumplir los datos almacenados en la base de datos.
- **Definición de vistas**: El LDD incluye comandos para definir las vistas.
- **Control de transacciones**: SQL tiene comandos para especificar el comienzo y el final de una transacción.
- **SQL incorporado y dinámico**: Esto quiere decir que se pueden incorporar instrucciones de SQL en lenguajes de programación como: C++, C, Java, Cobol, Pascal y Fortran.
- **Autorización**: El LDD incluye comandos para especificar los derechos de acceso a las relaciones y a las vistas.

Java

Es uno de los lenguajes de programación más potentes y populares actualmente. Fue desarrollado por James Gosling de Sun Microsystems (ahora adquirida por Oracle). Su sintaxis deriva de C y C++ aunque no llega hasta tan bajo nivel.

Las aplicaciones Java se ejecutan en una máquina virtual denominada JVM, de forma que el código se abstrae de la arquitectura que tiene debajo, dejando ese trabajo a la máquina virtual. El lenguaje Java es de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado para que hubiera las menos dependencias de implementación como fuera posible. Las aplicaciones se compilan a un código llamado bytecode (clases de Java) que son las que posteriormente se ejecutan en las máquinas virtuales.

Se puede ver un sencillo código en Java que nos muestre el "Hola Mundo" en un Applet:

```
// Hello.java
import javax.swing.JApplet;
import java.awt.Graphics;

public class Hello extends JApplet {

       public void paint(Graphics g) {
        g.drawString("Hola, mundo!", 65, 95);
    }

}
```

Figura. 4.5. Ejemplo de código en Java

Java es el lenguaje de programación de aplicaciones para el sistema operativo Android. La manera de desarrollar aplicaciones para este SO es algo diferente a la de implementar aplicaciones estándar de Java, pero más bien supone una extensión del propio Java S.E. De esta forma Android nos provee de toda la potencia de Java aplicada al desarrollo de aplicaciones para este entorno.

Android

Es un sistema operativo basado en Linux y fue diseñado principalmente para dispositivos móviles con pantalla táctil como los teléfonos inteligente o tabletas. Desde sus inicios Google financió y posteriormente compró la compañía que es el producto principal de la Open Handset Alliance, un conglomerado de fabricantes y desarrolladores de hardware y software y operadores de servicio.

La estructura del sistema operativo Android se compone de aplicaciones que se ejecutan en un framework Java de aplicaciones orientadas a objetos sobre el núcleo de las bibliotecas de Java en una máquina virtual Dalvik con compilación en tiempo de ejecución. Las bibliotecas escritas en lenguaje C incluyen un administrador de interfaz gráfica (surface manager), un framework OpenCore, una base de datos relacional SQLite, una Interfaz de programación de API gráfica OpenGL ES 2.0 3D, un motor de renderizado WebKit, un motor gráfico SGL, SSL y una biblioteca estándar de C Bionic. El sistema operativo está compuesto por 12 millones de líneas de código, incluyendo 3 millones de líneas

de XML, 2,8 millones de líneas de lenguaje C, 2,1 millones de líneas de Java y 1,75 millones de líneas de C++.

Android cuenta con las siguientes características:

- Diseño adaptado al dispositivo.
- Almacenamiento en SQLite.
- Conectividad con GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, HSDPA, HSPA+ y WiMAX.
- Navegador web incorporado.
- Soporte java para la ejecución de aplicaciones.
- Soporte multimedia de los siguientes formatos: WebM, H.263, H.264 (en 3GP o MP4), MPEG-4 SP, AMR, AMR-WB (en un contenedor 3GP), AAC, HE-AAC (en contenedores MP4 o 3GP), MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF y BMP.
- Soporte para streaming.
- Soporte para hardware adicional como cámaras de fotos, pantallas táctiles, acelerómetros, giroscopios, etc.
- Entorno de desarrollo con emulador de dispositivos, herramientas para depuración y análisis de rendimiento.
- Multi-táctil.
- Bluetooth
- Multitarea real de aplicaciones.
- Características basadas en voz.
- Tethering permitiendo utilizar al dispositivo como punto de acceso.

JavaScript

JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas aunque existe una forma de JavaScript del lado del servidor.

JavaScript se diseñó con una sintaxis similar al C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo Java y JavaScript no están relacionados y tienen semánticas y propósitos diferentes. Fue desarrollado originalmente por Brendan Eich de Netscape con el nombre de Mocha, el cual fue renombrado posteriormente a LiveScript, para finalmente quedar como JavaScript.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

En el siguiente ejemplo de código se muestra la función recursiva de factorial en JavaScript:

```
function factorial(n) {
    if (n === 0) {
        return 1;
    }
    return n * factorial(n - 1);
}
```

Figura. 4.6. Ejmplo código Javascript

JQuery

Es una biblioteca de código JavaScript que permite de manera sencilla interactuar con los documentos HTML, manipular su DOM e integrar AJAX. Es un software libre y de código abierto bajo la licencia del MIT y GNU v2.

Se utiliza como una potente herramienta que permite a los desarrolladores crear efectos atractivos y dinámicos en las páginas web de una manera rápida y limpia. Cuenta con las siguientes características:

- Selección de elementos DOM.
- Interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3 y un plugin básico de XPath.
- Eventos.
- Manipulación de la hoja de estilos CSS.
- Efectos y animaciones.
- Animaciones personalizadas.
- AJAX.
- Soporta extensiones.
- Utilidades varias como obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes, etc.
- Compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari 3+, Opera 10.6+ y Google Chrome 8+.5

5.- El robot Karotz.

La fase número dos del proyecto presenta todas las características del robot Karotz, las interfaces de comunicación con los usuarios y su máquina virtual.

5.1.-Introducción.

Karotz es un robot de la firma francesa Aldebarán Robotics. Esta compañía fabricante de los robots Nao adquirió recientemente estos pequeños *conejitos*. El origen de Karotz proviene de la empresa Violet cuando comenzó a desarrollar los *Nabaztag* (que significa liebre en armenio). Son considerados dispositivos electrónicos de comunicación y presentan un aspecto similar al de un conejo.

El propósito final de este sistema es proporcionar al usuario una serie de actividades que le parezcan interesantes de manera eficaz y atractiva. Dentro de estos últimos aspectos entra el robot Karotz. Su introducción en el sistema permite generar un interés mayor en los usuarios.

Imagen 5.1. Robot Karotz de frente

La implementación del módulo cliente de recomendación de actividades ha sido diseñada para aprovechar las características del robot. En el siguiente apartado se describirá todas las funcionalidades con las que cuenta este dispositivo y cómo se aprovechan para el sistema de recomendación de actividades.

5.2.-Análisis del robot.

5.2.1.- Características del robot.

El robot Karotz presenta una forma similar a la de un conejo gracias al par de orejas desmontables con el que viene equipado.

Entre sus características y funcionalidades principales nos encontramos las siguientes:

- Transmisión de datos por internet.
- Comunicación mediante el habla.

Estas dos características son claves para el propósito del robot en este sistema.

Vista frontal de Karotz

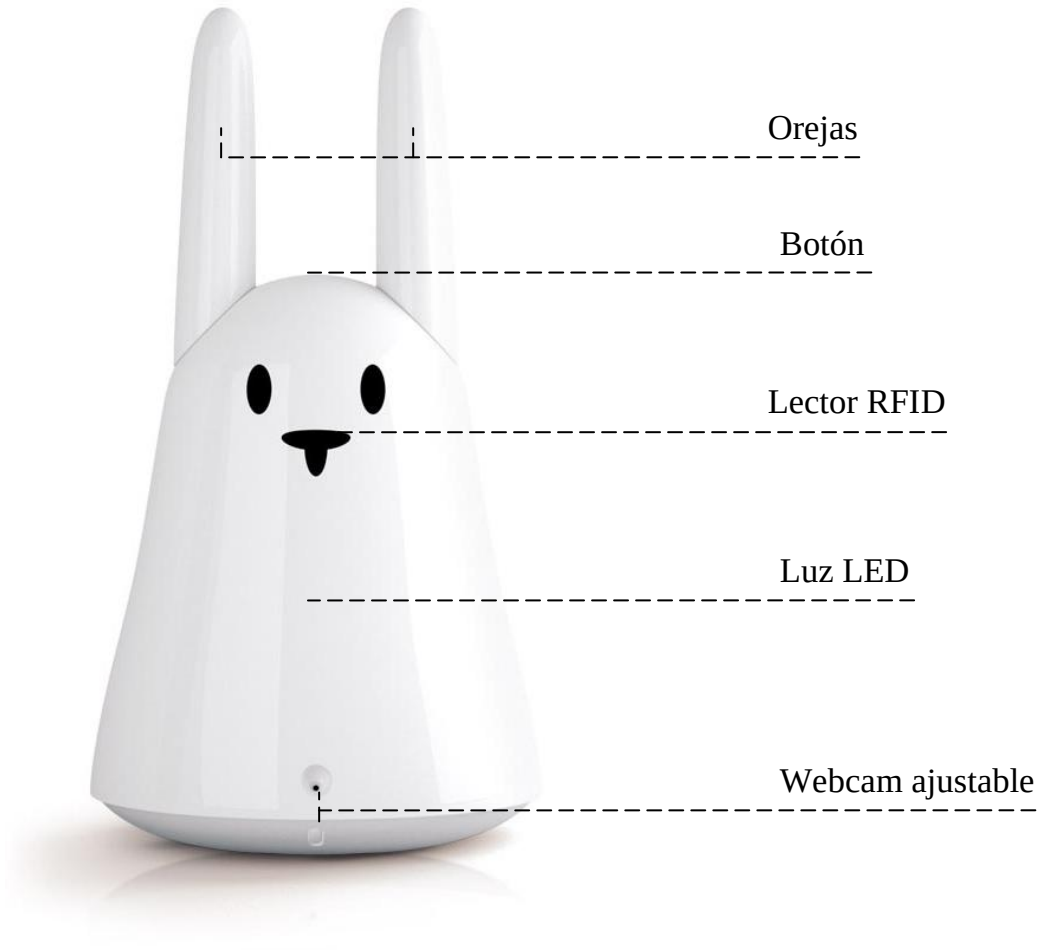


Imagen 5.2. Partes frontales del Robot Karotz

Vista trasera de Karotz

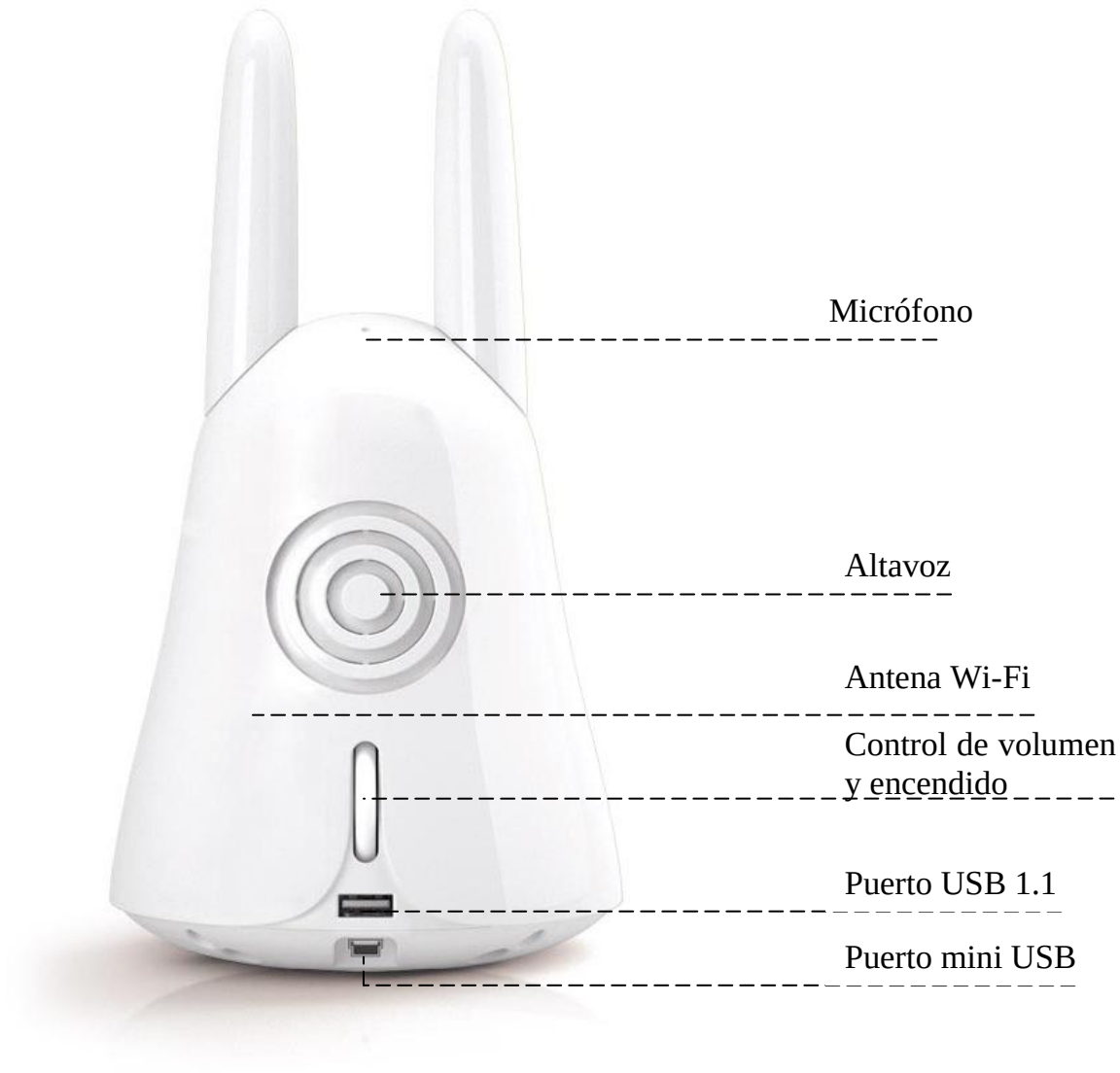


Imagen 5.3. Partes traseras del robot Karotz

Flujo de puesta en marcha de Karotz

Karotz para iniciarse siempre debe conectarse al servidor de Aldebaran. A continuación se queda en modo espera hasta que se le inicie alguna de sus aplicaciones (módulo cliente de recomendación de actividades) mediante la API REST, el botón o por RFID.

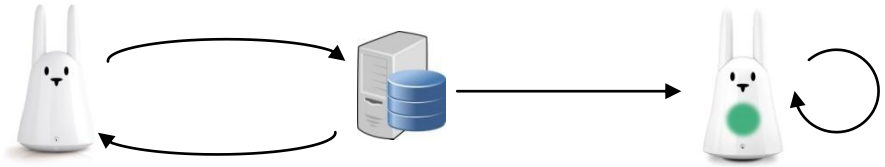


Figura. 5.1. Flujo de puesta en marcha de Karotz

Características técnicas:

Procesador:
ARM9 Processor de 400Mhz. Este procesador está basado en la arquitectura ARMv5TE. Consta de un pipeline de 5 etapas (Fetch/Decode/Execute/Memory/Writeback) y 31 registros de 32 bits. Soporta los conjuntos de instrucciones ARM y Thumb.

Memoria:
- 64MB de memoria RAM.
- 256MB de memoria Flash

Interfaces I/O:
- Micrófono integrado
- Altavoz integrado
- Webcam
- Antena Wi-Fi
- Lector RFID
- 1 puerto USB 1.1
- 1 puerto mini USB

Dimensiones:
- Altura: 23cm.
- Peso: 418g.

Compatible con:
- Windows
- Mac
- Linux

Motores:
- 2 servo-motores que hacen girar las orejas.
- Rotación de las orejas desmontables en 360 grados

Idiomas:
- Español
- Francés
- Inglés
- Alemán

Software:
Karotz posee un sistema basado en el Linux con kernel 2.6, uClibC y BusyBox. A esta base se le han añadido librerías de sistema, multimedia, web y otras.

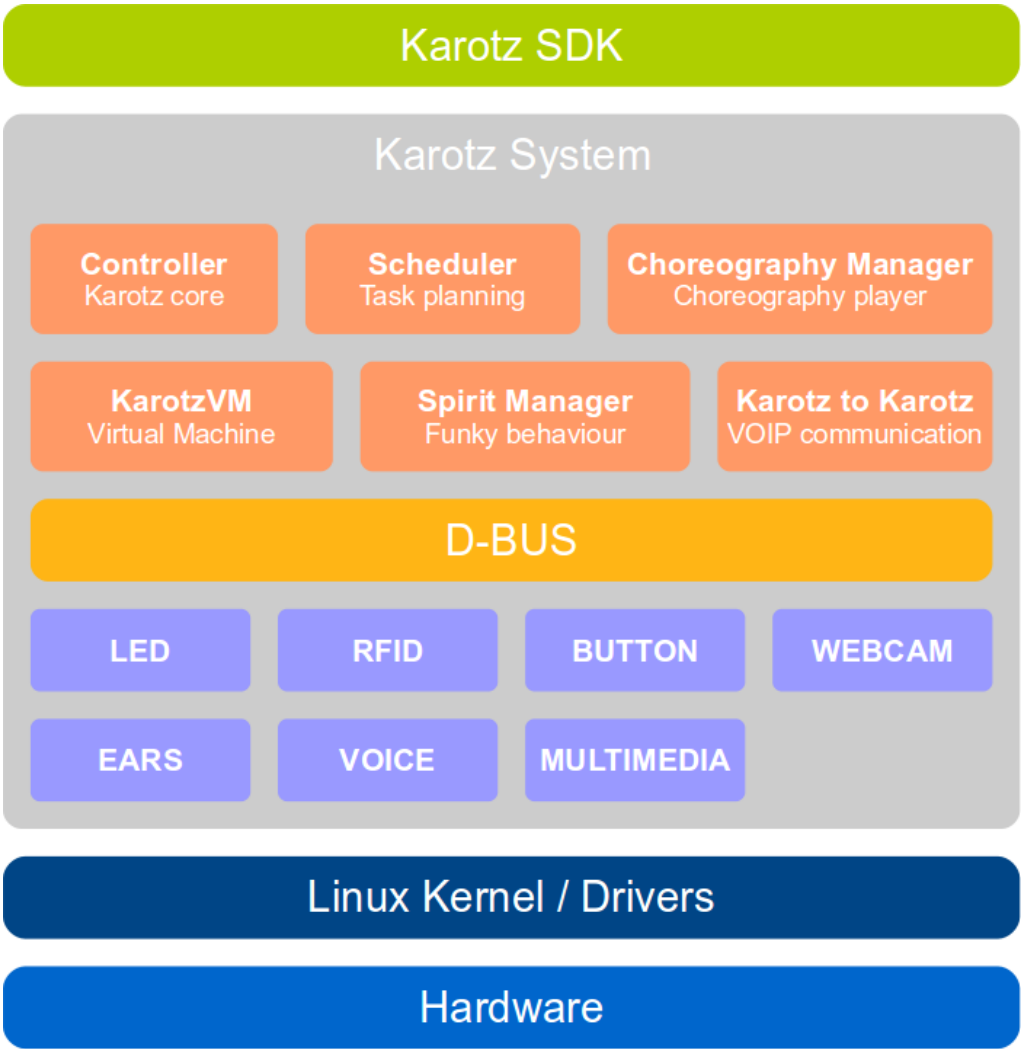


Figura. 5.2. Arquitectura de componentes en Karotz

Contiene diferentes módulos independientes que se comunican unos con otros a través del sistema de comunicación D-Bus. La conjunción de todos ellos da como resultado el sistema operativo de Karotz.

Módulos principales:

- El controlador (Controller): es el corazón de Karotz, centraliza todos los datos que el recibe desde fuera hacia el servidor, y supervisa el funcionamiento general del robot.
- Karotz to Karotz: es el módulo de comunicación por VOIP entre Karotz.
- La agenda (The Scheduler): permite planificar la ejecución de las tareas y las aplicaciones JavaScript en el momento preciso (por ejemplo despertar al usuario a las 7:00).

- The Spirit Manager: gestiona los diferentes modos de comportamiento del conejo.

Módulos I/O:

- RFID: Detección y lectura de chips RFID.
- Voice: Reconocimiento de voz y síntesis del habla.
- Multimedia: Lectura de ficheros multimedia de la memoria flash del propio conejo, del USB o a través de streaming (http, rtsp, mms).
- Webcam: Permite tomar fotografías cuando se le hace una petición.
- Button: Detecta los diferentes tipos de pulsaciones.
- LED: Controla los LEDs de Karotz. Los colores, la intensidad y la velocidad de los pulsos.
- Ears: Gestiona el control de la orejas y el detector de movimiento.

5.2.2.- Aplicación *Karotz Controller*.

Karotz Controller es una aplicación que permite tener el control de varias características de Karotz de manera remota. Funciona en iPad, iPod Touch, iPhone y dispositivos basados en Android.

Imagen 5.4. Aplicación Karotz Controller

Tiene las siguientes funcionalidades:

- Ver lo que Karotz ve gracias a su webcam.
- Tomar fotografías.
- Enviar mensajes de texto o audio que Karotz dirá.
- Cambiar la posición de las orejas.
- Reproducir música.
- Reproducir radios web.
- Cambiar el color de los LEDs.

5.2.3.- Análisis de la API de trabajo de Karotz.

Karotz provee a los programadores dos caminos distintos para desarrollar sus programas y controlar al robot. El primero es una API REST y el segundo, que es por el que se ha optado, es una API en JavaScript que embebe el programa desarrollado dentro robot. Además, este último método permite testear los programas con ejecución directa, utilizando KarotzSDK y una conexión USB.

Métodos genéricos:

setTimeout(int : duracion, function : callback)
Este método permite esperar un tiempo antes de ejecutar la función que se llama en el callback. Por ejemplo: setTimeout(5000,myfunction) espera cinco segundos antes de ejecutar *myfunction*.

http.get(url : url)
Obtiene el contenido de la url especificada.

http.get2(url: url)
Obtiene el contenido de la url especificada y la cabecera de la respuesta.

http.post(url : url, string : datos, string : cabecera, Boolean : es multiparte);
Envía los datos y la cabecera a la url especificada y devuelve el resultado de la acción.

file.read(string : nombre del fichero)
Permite leer el contenido de un fichero que se encuentre en Karotz o un USB. El fichero debe ser de texto y devuelve el contenido leído.

file.write(string : nombre del fichero, string : datos)
Escribe en un fichero de texto el texto pasado en el segundo argumento.

exit()
Es el método que finaliza el programa que se encuentra en ejecución.

Métodos de Karotz:

Listeners:
Los listeners son funciones que están ejecutándose en background y saltan a primer plano cuando detectan un evento que están manejando. Estos eventos pueden ser: pulsar un botón, pasar un RFID, mover las orejas, eventos multimedia y eventos del puerto serie.

karotz.button.addListener(function(event))
Utilizando esta función en nuestro programa principal es posible interrumpirlo y detectar qué tipo de pulsación se está realizando.

karotz.rfid.addListener(function(data))
El listener del RFID proporciona todos los datos del chip RFID leído (id, referencia a una aplicación, color, tipo y dirección).

karotz.ears.addListener(function(event,step,length))
Con este listener es posible detectar si se ha movido alguna de las orejas y cuál de ellas ha sido. Es capaz de indicar el tiempo que ha llevado y el número de pasos que se ha girado la oreja, teniendo un total de 17 por ciclo.

karotz.multimedia.addListener(function(event))
Detecta si el reproductor multimedia se ha iniciado, pausado, parado o actualizado.

karotz.serial.addListener(function(data))
Permite detectar y obtener los datos que se están enviando por el puerto serie.

karotz.ping()
Este método envía un ping a Karotz y si no responde pasados 15 minutos la aplicación finalizará automáticamente.

karotz.tts.start(string : texto, string idioma, function : callback)
Con este método (Text To Speech) es posible hacer que Karotz hable lo que se le pasa por el primer parámetro. El segundo corresponde al idioma (fr, en, de, es) y el tercero a la función que se llamará como callback.

karotz.asr.string(string gramática, string idioma, function(asrResult))
El método (Automated Speech Recognition) permite que se reconozca lo que se le está diciendo a Karotz. Para ello como primer parámetro se le indica una gramática, por ejemplo: “0 | (uno | primero) | 2 | tres | 4 | 5 | 6 | siete [8 | 9]”. Como segundo parámetro el idioma en el que debe reconocer las sentencia y el tercer parámetro debe ser el callback.

karotz.ears.move(int : izquierda, int : derecha, function callback)
Mueve las orejas tantos pasos como se le indica módulo 17, es decir, entre 0 y 16.

karotz.ears.moveRelative(int : izquierda, int : derecha, function callback)
Mueve las orejas con respecto a la posición relativa que tienen.

karotz.ears.reset(function callback)
Pone las orejas en su posición inicial.

karotz.led.light(string : color)
Cambiar el color del LED de Karotz. El parámetro es una string en formato hexadecimal con valores RGB, por ejemplo "FF0000" es rojo.

karotz.led.fade(string : color, int : tiempo, function callback)
Cambiar el color actual de robot al nuevo indicado de forma suavizada.

karotz.led.pulse(string : color, int period del pulso,int : duración total , function callback)
Hace que el LED parpadee con una frecuencia y una duración determinada.

karotz.webcam.photo(string : url)
Al ejecutarse este método se toma una fotografía con la cámara web de Karotz y se envía a la dirección url indicada por parámetro.

karotz.multimedia.play(string : ruta, function callback)
Reproduce un fichero multimedia que debe estar en la ruta indicada dentro de un USB.

karotz.multimedia.pause(function callback)
Pausa la reproducción activa en ese momento.

karotz.multimedia.resume(function callback)
Continúa reproduciendo el fichero que se quedo en *pause*.

karotz.multimedia.stop(function callback)
Para la reproducción multimedia en curso.

karotz.multimedia.record(function(event))
Permite la grabación en un fichero con formato *flac* que se debe enviar a un servidor externo por internet.

karotz.multimedia.next(function(event))
Reproduce la siguiente canción.

karotz.multimedia.previous(function(event))
Reproduce la canción anterior.

karotz.multimedia.artist(function(event))
Devuelve la lista de canciones para el artista indicado

karotz.multimedia.folder(function(event))
Devuelve la lista de directorios en el USB.

karotz.multimedia.playlist(function(event))
Devuelve las listas de reproducción que se encuentran en el USB.

karotz.multimedia.genre(function(event))
Devuelve una lista de los géneros presentes en el USB.

karotz.multimedia.song(function(event))
Devuelve una lista de todas las canciones del USB.

karotz.serial.open(port,baudrate)
Abre un puerto especificado a la velocidad indicada. El puerto normal USB es "/dev/ttyUSB0" y el puerto mini USB es "/dev/ttyGS0". Los valores que acepta de velocidad son: 50, 75, 110, 134, 150, 200, 300, 600,1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400 y 460800.

karotz.serial.write(data)
Este método permite escribir datos ASCII

karotz.serial.writehexa(data)
Este método permite escribir datos en hexadecimal.

5.3.-Comunicación con el robot.

Como ya se ha comentado anteriormente el Karotz provee diversas formas para comunicarse con los usuarios. Algunas de estas formas están implementadas como *listeners* (detección de movimiento de las orejas, botón) y otras como métodos.

Para este proyecto la interfaz que más interesa y que servirá profundamente a los usuarios es el habla. Por ello se dedica el siguiente apartado a definir claramente cómo Kartoz interactúa utilizando esta interfaz. La otra opción útil es el uso de RFID para casos concretos en la aplicación.

5.3.1.- Comunicación por voz.

La comunicación por voz se divide en dos Text To Speech (TTS) y Automated Speech Recognition (ASR).

Text To Speech
Así es como se conoce la generación de texto a voz. Se utiliza una voz artificial de la que ya se tienen sílabas o letras almacenadas y posteriormente se van juntando para formar palabras y frases.

Fases para convertir texto a voz:

1. **Normalización del texto**. Se convierte la totalidad del texto a una forma textual convencional. Esto afecta principalmente a las cifras, abreviaturas, etc. A la normalización del texto también se la denomina pre-procesado o tokenización.
2. **Conversión fonética**. Una vez normalizado el texto se asignan transcripciones fonéticas a cada palabra. El proceso de convertir las transcripciones fonéticas en palabras se denomina conversión texto-fonema.
3. **División prosódica**. Se divide el texto en unidades prosódicas, tales como unidades sintagmáticas, proposiciones y frases.

Se puede ver un ejemplo de cómo Karotz hace TTS con la siguiente porción de código:

```
var  text = “Hola Mundo”;
karotz.TTS.start(text, 'es', function(event){
  if ((event = "TERMINATED") || (event == "CANCELLED")) {
    karotz.multimedia.play(radiourl);
  }
})
```

Figura. 5.3. Ejemplo de código en Karotz haciendo TTS

Automated Speech Recognition
Este es uno de los grandes desafíos en el campo de la computación aún hoy en día y supone todo un reto para el futuro. Actualmente se están intentando crear algoritmos reconocimientos de voz eficaces aplicando Cadenas Ocultas de Markov, redes neuronales o *dynamic time warping*.

El reconocimiento de voz utiliza una gramática ABNF, que es un tipo de gramática basada en una BNF pero tiene su propia sintaxis y reglas de derivación.

Ejemplo de gramática ABNF:

```
[[very] big] pizza ([with | and] $topping)*
```

Figura. 5.4. Ejemplo de gramática ABNF

Karotz realiza un *matching* entre el sonido que escucha y la gramática ABNF. El resultado será la frase o "<nomatch>" si no encuentra resultados.

Es posible observar un ejemplo de código del reconocedor de voz en el siguiente cuadro:

```
var gramatica = "Hola [Mundo] | adios | Karotz | Conejito";
karotz.asr.string(gramatica, "es", function(asrResult) {
  var he_oido = asrResult.text;
  if (he_oido == "<nomatch>")
      setTimeout(2000, function() { has_dicho() });
  else
      karotz.tts.start("Has dicho : " + he_oido, "es",
                       has_dicho);
});
```

Figura. 5.5. Ejemplo de código en Karotz haciendo ASR

Caracterización del micrófono/ASR
Karotz presenta problemas de comprensión como cualquier dispositivo reconocedor del habla. Estos problemas pueden ser debidos a varios factores:

- **Entorno muy ruidoso**. Karotz es capaz de entender al usuario en entornos que no se encuentran totalmente en silencio, pero le resulta mucho más complejo cuando el nivel de decibelios es superior a 30 o 40.
- **Distancia del usuario a Karotz**. Esta distancia no debe ser mayor de 50cm porque si no el micrófono de Karotz no captará bien el sonido.
- **Corrección en el uso de las palabras.** La entonación y corrección del hablante es fundamental para que el robot lo entienda.

5.3.2.- Comunicación y uso de RFID.

Otra de las vías interesantes para comunicarse con Karotz es utilizando la tecnología RFID (Radio Frequency IDentification). Con ella es posible identificar objetos ya que funciona como un sistema de almacenamiento y recuperación de datos. Los dispositivos que llevan incorporada esta tecnología guardan unas etiquetas con las que es posible ser identificados. Además no es necesario que se encuentren en visión directa para poder comunicarse.

Se pueden dividir en dos tipos de dispositivos, los activos y los pasivos. Las etiquetas pasivas no necesitan alimentación interna mientras que las activas sí.

En Karotz nos encontramos los Mini Karotz's que son etiquetas pasivas de RFID. Se distinguen en *flatnanoz* y *nanoztag*:

Imagen 5.5. Conjunto de Flatnanoz

Imagen 5.6. Conjunto de Nanoztag

Los *flatnanoz* y *nanoztag* pueden lanzar las aplicaciones que hayan sido asociadas a Karotz. Para ello es necesario seguir los siguientes pasos:

1. Frotar un Mini Karotz por la parte frontal de Karotz.
2. A continuación el objeto aparecerá disponible en la cuenta de Karotz.com
3. Pulsar sobre unir el objeto a Karotz.
4. Hacer clic sobre la aplicación que se quiere asociar.
5. Seleccionar el dispositivo RFID que se desea unir.

Otra posibilidad es utilizar el *listener* de la API de Karotz. Este método nos proporciona los siguientes atributos del chip RFID:

- data.id: El identificador único del chip.
- data.direction: 1 si se está leyendo, 2 si se deja de leer.
- data.app: No es usado.
- data.pict: No es usado.
- data.type: es NONE_TYPE, FLAT, NANOZTAG, ZTAMPS

- data.color: es NONE_COL ; RED ; BLUE ; GREEN ; YELLOW ; PINK ; BLACK ; GREY ; ORANGE ; PURPLE ; WHITE ; DARK_RED ; DARK_BLUE ;

De esta manera, dentro de una aplicación ya ejecutada en Karotz, es posible utilizar los atributos que proporciona para realizar tareas dentro con chips RFID.

En el siguiente cuadro se muestra un ejemplo:

```
karotz.rfid.addListener(myrfid);

var tag_type, tag_id, tag_color, tag_direction;
var myRfid =function(data) {
   tag_id = data.id;
   tag_app = data.app;
   tag_color = data.color;
   tag_type = data.type;
   tag_direction = data.direction;
}
```

Figura. 5.6. Ejemplo de código en Karotz utilizando RFID

5.4.-KarotzVM.

Debido a que Karotz no tiene ni teclado ni pantalla ni módulo de programación, se provee a los desarrolladores de una máquina virtual denominada KarotzVM. Esta máquina está programada en Java y viene empaquetada en un fichero JAR. Con ella es posible testear los programas emulando su ejecución en el robot o ejecutarlos directamente en Karotz utilizando el cable USB o por internet.

6.- Sistema de recomendación.

6.1.-Introducción.

Probablemente todos hemos pasado por motores de recomendación en sitios de compras online como Amazon. Los sistemas de recomendación son la aplicación de técnicas y metodologías para obtener un producto final personalizado. La mayoría de estos sistemas llevan en su núcleo un algoritmo que puede ser entendido como una instancia particular de *Data Mining*.
El proceso de *data mining* se divide en tres pasos sucesivos: preprocesado de datos, análisis de datos e interpretación.

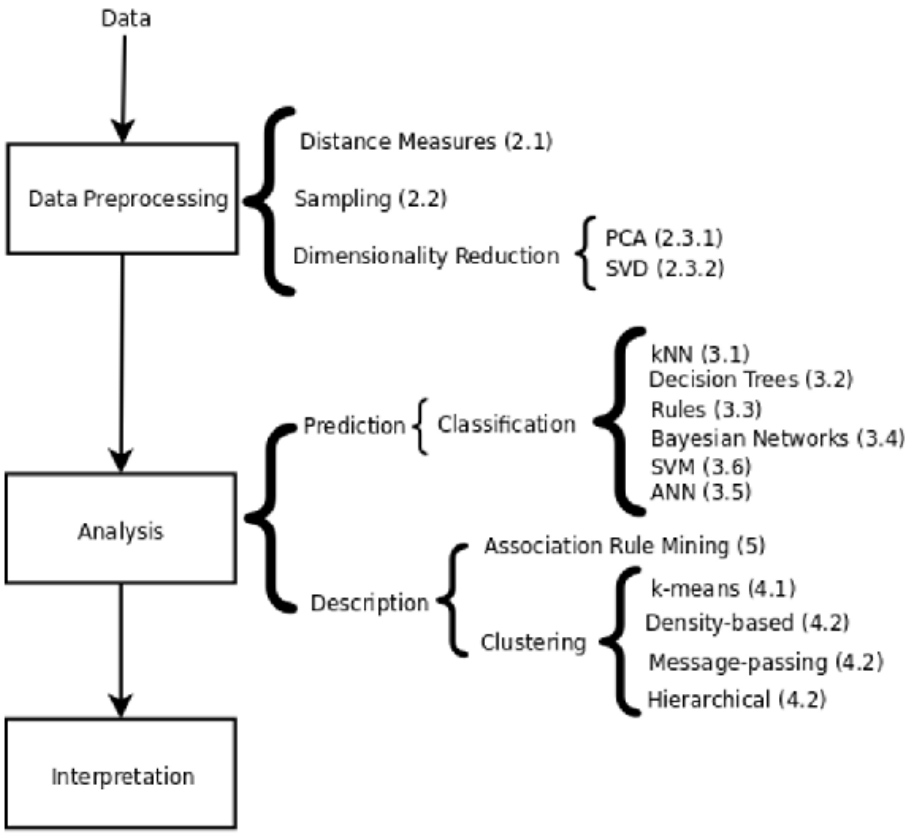


Figura. 6.1. Esquema de recomendación

Se puede definir datos como una colección de objetos y sus atributos, donde un atributo está definido como una propiedad o característica de un objeto.

El preprocesado se encarga de filtrar, limpiar y transformar los datos de entrada de manera que a continuación puedan ser clasificados o descritos. Para ello tenemos la medidas de similitud, el muestreado y la reducción de dimensión.

La procesado y clasificación es el mapeo entre un espacio de componentes y un espacio de nombres, donde los componentes representan un espacio de

características de objetos a clasificar y el espacio de nombres representan las clases.

La interpretación representa el significado final obtenido del conjunto de datos tras ser procesado, de forma que represente un elemento final, como por ejemplo una recomendación.

6.2.-Técnicas de recomendación.

En los sitios de recomendación se realiza un seguimiento de los hábitos de compra de todos sus compradores y cuando se loguean en el sitio se usa esta información para realizar sugerencias de productos.

Existen gran cantidad de preferencias que pueden ser almacenadas en diferentes formas. En ocasiones los datos son los productos que el usuario ha comprado junto con opiniones representadas por votos en rango 1-5.

Las técnicas de recomendación se dividen en cuatro:

- Recomendación colaborativa o filtrado colaborativo (CF): La recomendación colaborativa está basada en la premisa de que si has compartido en el pasado intereses con algún usuario, en el futuro tendrás los mismos gustos. Esta es la principal técnica empleada que abordaremos más adelante. La idea es utilizar lo que dice gente similar a ti para recomendarte elementos. Es la técnica más utilizada para recomendación de productos.
- Recomendación basada en contenido (CB): La recomendación basada en contenido trata de aprender de las preferencias del usuario. De acuerdo a ellas, busca aquellos elementos que le son similares a los gustos del usuario. La mayoría de estas técnicas están aplicadas a la recomendación de textos.
- Recomendación basada en conocimiento: Se utiliza cuando existe un número demasiado bajo de valoraciones y está basado en casos y restricciones.
- Recomendación híbrida: mezcla los tres conceptos anteriores para ofrecer al usuario la mejor experiencia.

6.2.1.- Filtros Colaborativos (CF).

Los filtros colaborativos representan la técnica más empleada ya que ofrece rendimiento con buenos resultados. El filtro se establece por los vecinos más próximos (kNN, k nearest-neighbors) Este tipo de técnicas están basadas en la idea de que gente similar a ti a la que le gusto algo hará que a ti te guste. Para

conocer que le gusta a la gente es necesario que realice votaciones y obtener un conjuntos de datos.

Recolección de preferencias

Para la recolección de preferencias de los usuarios se puede utilizar una lista de votaciones de productos. Esta lista de votaciones supone la matriz de entrada, la matriz de usuarios-elementos, con la que se trabajará durante todo el proceso de recomendación.

Entrada

- Conjunto *U* de usuarios, usuarios clientes por ejemplo.
- Conjunto *I* de ítems, como por ejemplo actividades.
- Conjunto totalmente ordenado *R* de valores para votar, por ejemplo {1,2,3}
- Relación funcional $r: U \times I \rightarrow R$
- $r(u, x)$ representa la valoración del usuario *u* por el item *x* en la escala *R*

```
Carlos = {"Ciclismo Canarias" : 3, "Rapel Canarias" : 2, "Buceo
Marismas" : 1}

María = {"Ciclismo Canarias" : 2, "Rapel Canarias" : 2, "Buceo
Marismas" : 3}

Juan = {"Ciclismo Canarias" : 1, "Rapel Canarias" : 3, "Buceo
Marismas" : 1}

Luis = {"Ciclismo Canarias" : 2, "Rapel Canarias" : 2, "Buceo
Marismas" : 3}
```

Figura. 6.2. Ejemplo de votaciones de actividades

Buscando usuarios similares

El siguiente paso consiste en determinar cuánto de similares son los usuarios y sus votaciones. Para ello podemos utilizar varios métodos:

Distancia Euclídea

Es el método más sencillo y más común. Resulta óptimo cuando las valoraciones se encuentran normalizadas (ej. votaciones de 1-3 estrellas).

$$sim(x,y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

Figura. 6.3. Distancia Euclídea

donde *n* es el número de dimensiones (atributos) y x^k y y^k son el k^{th} atributos (componentes) de los objetos de datos x e y respectivamente.

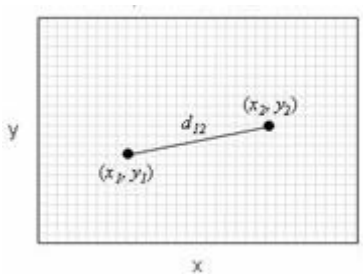


Figura. 6.4. Ejemplo de Distancia Euclídea

Distancia de Minkowski

La distancia Minkowski representa una generalización de la Distancia Euclídea:

$$sim(x,y) = (\sum_{k=1}^{n} |x_k - y_k|^r)^{\frac{1}{r}}$$

Figura. 6.5. Distancia de Minkowski

donde *r* es el grado de la distancia. Dependiendo del valor de r la distancia Minkowski toma diferentes nombres. Para r = 1, Distancia Manhattan, norma L1; para r = 2, Distancia Euclídea; para r = ∞, la Distancia suprema.

Distancia de Mahalanobis

Se define como:

$$sim(x,y) = \sqrt{(x-y)\sigma^{-1}(x-y)^T}$$

Figura. 6.6. Distancia de Mahalanobis

donde σ es la covarianza de la matriz de datos.

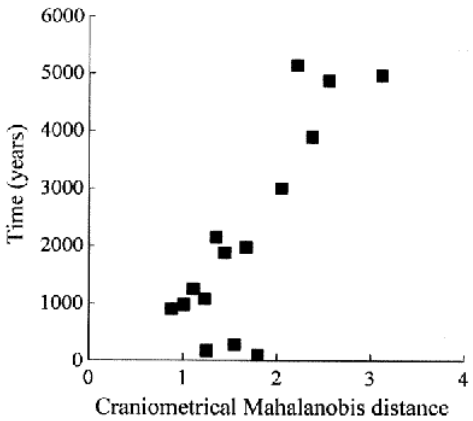


Figura. 6.7. Ejemplo de distancia de Mahalanobis

Distancia basada en el coseno

Es un enfoque muy común es considerar los elementos como un plano de vectores de un n-dimensional espacio y calcular su similitud como el coseno del ángulo que ellos forman:

$$sim(\vec{i},\vec{j}) = \frac{\vec{i} * \vec{j}}{|\vec{i}| * |\vec{j}|}$$

Figura. 6.8. Similitud basada en el coseno

donde * indica el producto de vectores y |i| es la normal del vector i.

Correlación de Pearson

La similitud entre elementos también puede ser dada por su correlación, que mide la relación lineal entre objetos. Aunque varios coeficientes de correlación que pueden ser aplicados, el Coeficiente de Pearson es el más común.

$$sim(x,y) = \frac{\sum_{k=1}^{n} x_k y_k - \frac{\sum_{k=1}^{n} x_k \sum_{k=1}^{n} y_k}{n}}{\sqrt{(\sum_{k=1}^{n} x^2 - \frac{(\sum_{k=1}^{n} x)^2}{n})(\sum_{k=1}^{n} y^2 - \frac{(\sum_{k=1}^{n} y)^2}{n})}}$$

Figura. 6.9. Correlación de Pearson

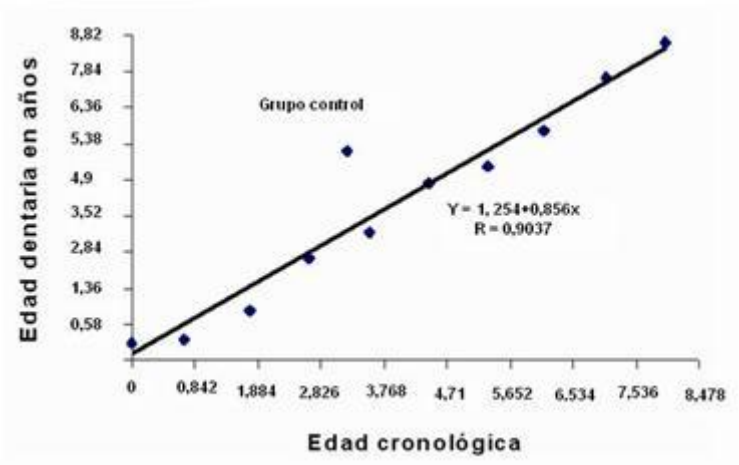


Figura. 6.10. Ejemplo de Correlación de Pearson

Realizando el ranking de los ítems

Una vez que se obtiene la similitud con el resto de los usuarios debemos hacer un ranking de los ítems basado en las valoraciones de esos usuarios. Para ello se va a tener en cuenta cual de los ítems es el que obtiene mejor puntuación basado en

las similitudes con los usuarios. La siguiente función nos da la puntuación para un ítem dado en función de las valoraciones y similitudes de los usuarios:

$$score(s,r) = \frac{\sum_{i=1}^{n} s_i r_i}{\sum_{i=1}^{n} s_i}$$

Figura. 6.11. Puntuación de un ítem basado en similitud

donde *s* y *r* son la similitud y votación respectivamente del usuario *i*.

Finalmente, para obtener el ítem a recomendar sólo es necesario elegir el de mayor puntuación:

$$item_{final} = \max(score)$$

Figura. 6.12. Ítem a recomendar

6.3.-Sistema de recomendación implementado.

El sistema de recomendación implementado es una versión híbrida de CF, CB y basada en conocimiento. De modo que siguiendo el esquema de un sistema de recomendación, inicialmente en el preprocesado se utilizan las técnicas de CB y recomendación basada en conocimiento junto con la determinación de la Distancia Euclídea entre los usuarios, CF. A continuación en el análisis utilizamos la clasificación por los kNN. Y finalmente obtenemos una salida que es la actividad a recomendar.

6.3.1.- Recolección de datos.

La recolección de datos del usuario se realiza mediante tres vías diferentes, que se presentan a continuación.

Visualizaciones del detalle de actividades
Cuando ocurre el evento de que un usuario se sienta atraído por el nombre y/o la descripción de una actividad y desee conocer más datos de la misma hace clic sobre el detalle de la actividad. Esta acción queda registrada de forma que es posible conocer cuantas veces se ha visualizado una actividad en detalle desde su creación.
A ese número de visualizaciones se le ha denominado *score (s)*. De forma que el conjunto de scores está formado por:

$$Scores = \{s \mid s \in N\}$$

Figura. 6.13. Conjunto de puntuaciones

Y tenemos tantas *Scores* como actividades existen registradas en el sistema:

Actividad	Score
$\boldsymbol{Actividad_1}$	$Score_1$
$\vdots$	$\vdots$
$\boldsymbol{Actividad_n}$	$Score_n$

Tabla 6.1. Ejemplo de scores de actividades

Ratings de las actividades

En el sistema desarrollado en este proyecto se ha implementado la funcionalidad de que los usuarios puedan votar las actividades. Estas votaciones son las que posteriormente nos permite realizar CF.

De forma que el espacio de valoraciones es el siguiente:

$$R_v = \{1,2,3\}\ y\ R_{vi} = \{r_{vi} \mid r_{vi} \in R_v\}$$

Figura. 6.14. Espacio de valoraciones

Y para nuestro sistema denominamos esas votaciones *rate (r)* como la tupla de tres elementos: actividad, usuario y votación:

$$Rate = \{(a_1, u_1, R_{v1}), (a_1, u_1, R_{v2}), \dots, (a_n, u_n, R_{vn})\}$$

Figura. 6.15. Conjunto de rate

Los usuarios pueden haber votado una actividad o no, por tanto a priori no existen los *ratings*. De esta forma tenemos como mínimo cero ratings y como máximo $a * u$ ratings, donde a y u representan el número de actividades y usuarios respectivamente.

Se puede definir una tabla como la siguiente de *ratings*:

Actividad	Usuario	Rating
$\boldsymbol{Actividad_i}$	$Usuario_m$	$Rating_a$
$\vdots$	$\vdots$	$\vdots$
$\boldsymbol{Actividad_j}$	$Usuario_n$	$Rating_b$
$\vdots$	$\vdots$	$\vdots$
$\boldsymbol{Actividad_j}$	$Usuario_m$	$Rating_c$

Tabla 6.2. Ejemplo de ratings de actividades

Preferencias de los usuarios

Para culminar la recolección de parámetros se ha realizado una categorización de actividades. Basando las preferencias de los usuarios sobre esas categorías es posible realizar una recomendación más acertada.

Los usuarios pueden elegir todas, algunas o ninguna de las categorías como preferencias. De esta forma en el sistema vamos a poder tener un máximo $u * c$ donde u y c representan los usuarios y las categorías respectivamente.

Se puede definir una tabla como la siguiente de preferencias:

Categoría	Usuario
$\boldsymbol{Categoría_i}$	$Usuario_m$
$\vdots$	$\vdots$
$\boldsymbol{Categoría_j}$	$Usuario_m$
$\vdots$	$\vdots$
$\boldsymbol{Categoría_j}$	$Usuario_n$

Tabla 6.3. Ejemplo de preferencias de usuarios

6.3.2.- Proceso de recomendación.

Búsqueda de actividades

Debido a que el problema tiene un crecimiento lineal que depende la cantidad de usuarios y la cantidad de actividades, se limitan los datos a procesar a fin de mejorar el rendimiento del sistema. Al realizar la limitación a *N* actividades, se seleccionan las *N* primeras ordenadas por *score*.

Para ello se filtrarán *N* actividades de acuerdo a las preferencias del usuario. En nuestro caso *N* = 100.

Recordando que las preferencias representan categorías, tenemos un conjunto P_u de preferencias del usuario tal que:

$$Categorías = \{c_1, c_2, \dots, c_n\}$$

Figura. 6.16. Conjunto de categorías

$$P_u = \{p \mid p \in Categorías\}$$

Figura. 6.17. Conjunto de preferencias del usuario

El conjunto de actividades A_u acordes al usuario seleccionado debe ser entonces el siguiente:

$$Actividades = \{a_1, a_2, \dots, a_n\}$$

Figura. 6.18. Conjunto de actividades

$$A_u = \{a \mid a \in Actividades \wedge categoría(a) \in P_u \wedge \max(score) \geq score(a) > \text{score}(posición(a_{n+1}))\}$$

Figura. 6.19. Conjunto de actividades aptas para el usuario

Búsqueda de usuarios

De nuevo con afán de reducir la dimensión del problema se restringe el número de usuarios a comparar similitudes a M.

Para ello es necesario, realizar un filtrado análogo al de las actividades, seleccionando usuarios que tengan preferencias similares a como nosotros.

Del conjunto de usuarios *U*, seleccionamos los *M* primeros usuarios que concuerden con algunas o todas nuestras preferencias para forma el conjunto de usuarios acorde U_u:

$$Usuarios = \{u_1, u_2, \dots, u_n\}$$

Figura. 6.20. Conjunto de usuarios

$$U_u = \{u \mid u \in Usuarios \wedge (preferencias(u) \cap P_u \neq \emptyset) \wedge u \neq u_{final}\}$$

Figura. 6.21. Conjunto de usuarios acordes

Búsqueda de ratings

Lo siguiente que debemos buscar son las votaciones de los usuarios seleccionados. Esas votaciones (*Ratings*) formarán el siguiente conjunto:

$$Ratings_i = \{r \mid r \in Rate \wedge usuario(r) \in U_u\}$$

Figura. 6.22. Conjuntos de ratings de usuarios

Por otro lado, también necesitamos el conjunto de votaciones para el usuario final R_u:

$$Ratings_u = \{r \mid r \in Rate \wedge usuario(r) \in U_{final}\}$$

Figura. 6.23. Conjuntos de ratings del usuario final

Comparación de elementos

En el diseño del sistema conocemos que es posible que un usuario haya votado una actividad pero otro no. Por tanto no tienen que tener las mismas actividades votadas o ni tan siquiera una. Para realizar correctamente el algoritmo de *Collaborative Filtering* es necesario que los elementos que se comparen sean los mismos, es decir, hayan votado, los mismos productos. Es por ello que este paso es esencial y consiste en realizar la unión de los conjuntos de votaciones del usuario final y el usuario i-ésimo a comparar. Para lo que nos creamos dos

nuevos conjuntos, uno para el usuario final RF_u y otro para el usuario a comparar RF_i. Podemos representarlos formalmente de la siguiente manera:

$$RF_u = \{r \mid r \in Ratings_u \wedge actividad(r) \in actividad(Ratings_i)\}$$

Figura. 6.24. Conjunto de ratings a comparar del usuario final

$$RF_i = \{r \mid r \in Ratings_i \wedge actividad(r) \in actividad(Ratings_u)\}$$

Figura. 6.25. Conjunto de ratings a comparar del otro usuario

Obtención de la similitud

Para la obtención de la similitud se ha elegido como método el cálculo de la Distancia Euclídea entre los usuarios. Aunque la *Correlación de Pearson* presenta mejores resultados, en este caso los *ratings* se encuentran normalizados y el tiempo de procesado es notablemente menor.

El algoritmo implementado para hallar la Distancia Euclídea entre dos puntos en el espacio con *n* atributos en PHP es el siguiente:

```
static function euclideanDistance($point1, $point2) {
    $sumSq = 0.0;

    for ($i = 0; $i < count($point1); $i++) {
        $sumSq += pow($point1[$i] - $point2[$i], 2);
    }

    return 1/(1 + sqrt($sumSq));
}
```

Figura. 6.26. Cálculo de la Distancia Euclídea normalizada en PHP

Si nos fijamos la salida del algoritmo ha sido normalizada para que ofrezca valores entre cero y uno. Cuando mayor sea el valor, mayor será la similitud. Calculándolo para todos los usuarios es como obtenemos un nuevo conjunto *S* formado por las tuplas de todos los usuarios y sus similitudes.

Obteniendo el ranking de actividades

Una vez calculada la similitud con cada usuario ya es posible hacer el ranking de las actividades basándonos en lo que voto cada uno de los otros usuarios y su similitud con el usuario final.

Ahora, para cada una de actividades aplicamos el algoritmo que fue descrito anteriormente. Podemos ver cómo queda implementado en PHP en el siguiente cuadro:

```
    static function itemRank($ranks, $sims) {
        $partialRank = 0.0;
        $simSum      = 0.0;

        for ($i = 0; $i < count($ranks); $i++) {
            $partialRank += $ranks[$i] * $sims [$i];
            $simSum      += $sims[$i];
        }

        if ($simSum > 0)
            return $partialRank/$simSum;
        else
            return -1;
    }
```

Figura. 6.27. Cálculo del ranking de actividades en PHP

Al aplicar este algoritmo a cada una de las actividades a recomendar se obtiene la actividad final con tan sólo hallar la actividad que tiene el máximo rate.

7.- El servidor.

7.1.-Especificación de requisitos de usuario.

7.1.1.- Descripción.

El proyecto consiste en ofrecer un servicio para los clientes de los hoteles. Este servicio muestra a los clientes las actividades del hotel y empresas de terceros que más le interesan, ya que se conoce su perfil de gustos.
Las actividades que se muestran al usuario pueden ser consultadas en detalle para conocer datos como: fecha de inicio, fecha de finalización, horario, periodo, precio e incluso localización.

7.1.2.- Glosario de conceptos.

- **Usuario cliente:** Un usuario es aquel que utiliza los clientes (robot, portal web o móvil) para visualizar las actividades o servicios de las empresas y hoteles. El usuario cliente tiene unas preferencias. Cuando se encuentre en un hotel registrado y lo desee puede unirse a ese hotel para recibir sus servicios y ver sus actividades.
- **Usuario hotel:** Un hotel es un usuario registrado en el sistema que representa a un hotel y puede ofrece actividades y realizar notificaciones a sus huéspedes. El usuario hotel puede tener actividades y noticias.
- **Usuario empresa:** Una empresa es un usuario que representa a una empresa y ofrece actividades. La empresa tiene actividades.
- **Actividad o servicio:** Una actividad o servicio es toda aquella actividad que tiene una fecha de comienzo, de fin, duración, repetición, precio, observaciones y otros datos relacionados.
- **Noticia:** Las noticias son textos que los hoteles pueden enviar a los usuarios unidos a su hotel.
- **Categoría:** Se entiende como una forma de etiquetar la actividad.

7.1.3.- Modelo del dominio.

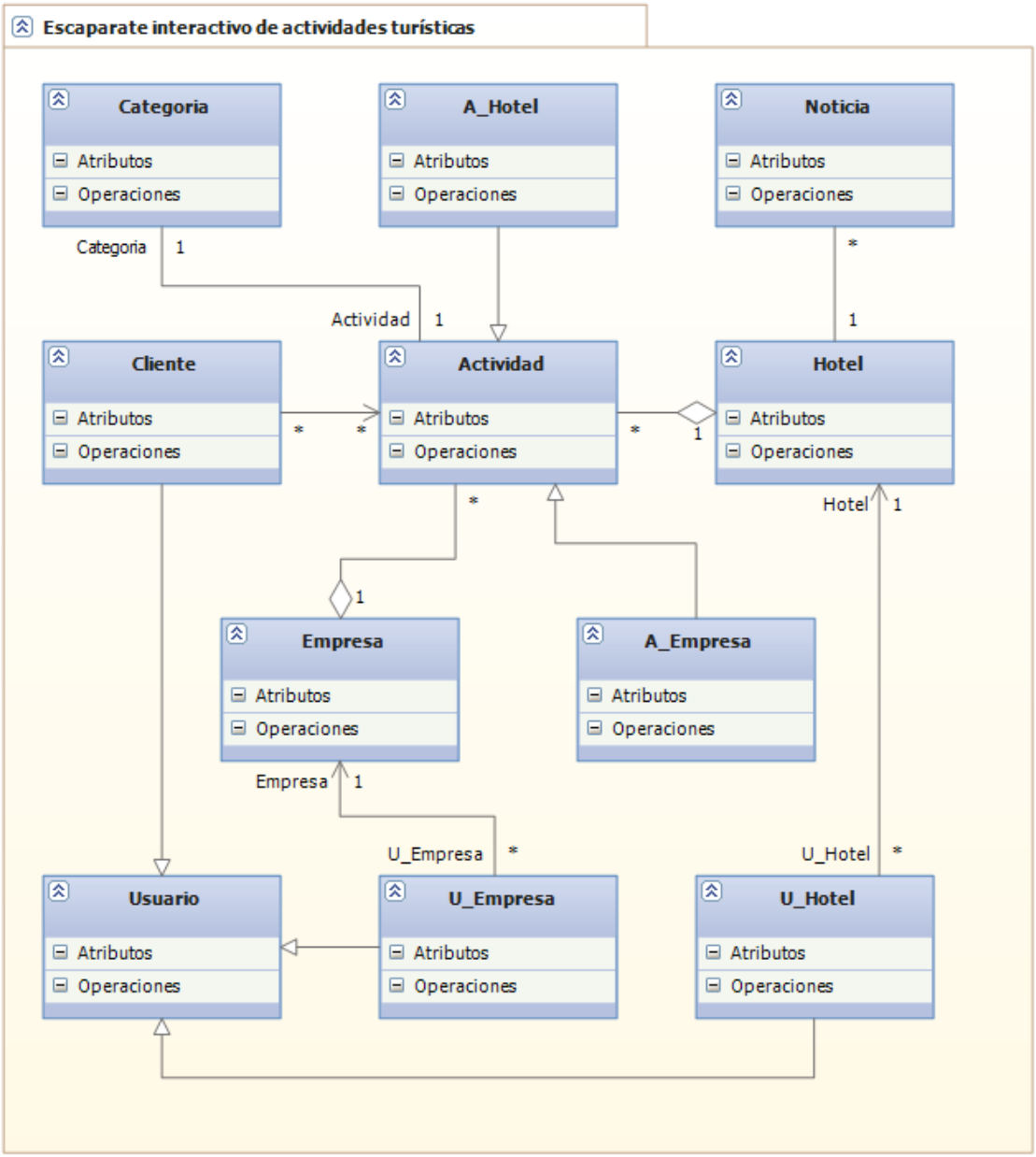


Figura. 7.1. Modelo de dominio

Donde podemos identificar las siguientes clases:

- Categoría: representa la categoría a la que pertenece una actividad.
- Cliente: es la clase que identifica al cliente que usa los servicios de visualizar actividades y recibir recomendaciones.
- Hotel: este elemento representa a los hoteles registrados en el sistema.
- Actividad: es una actividad ofertada por un hotel o por una empresa.
- A_Hotel: es un tipo de actividad ofertada por un hotel.
- A_Empresa: es una actividad ofrecida por una empresa.

- Noticia: es un elemento de texto informativo que los hoteles pueden mostrar a los usuarios.
- U_Hotel: Es el usuario que pertenece a un hotel y tiene la capacidad de añadir, modificar y borrar actividades y notificaciones de y para su hotel.
- U_Empresa: Es el usuario que pertenece a una empresa y tiene la capacidad de añadir, modificar y borrar actividades de y para su empresa.

7.2.-Especificación de requisitos software.

7.2.1.- Descripción.

El servidor del sistema almacena y gestiona todos los datos del sistema. El servidor esta subdividido en varias secciones porque ofrece servicios a distintos perfiles: usuarios empresa, usuarios hotel y clientes; además del de administrador del sistema.

Los hoteles son el servicio central por el cual se interrelaciona todo. Son registrados mediante un formulario en el sistema y pueden incluir datos como servicios propios y noticias, además pueden añadir otros usuarios que gestionará este administrador. Estos usuarios únicamente podrán realizar las acciones asociadas a este hotel. Además, los hoteles son quienes deciden qué actividades de otras empresas son las que promociona su hotel.

Las empresas se registran de manera análoga a los hoteles. Una vez registrados pueden ofertar detalladamente las actividades que tengan y gestionar usuarios que publiquen actividades de esa empresa.

Los usuarios clientes que deseen recibir recomendaciones personalizadas y unirse a hoteles también deben registrarse en el sistema. Deben indicar sus preferencias en cuanto a los tipos de actividades. El sistema, en función de sus gustos debe recomendarle actividades y servicios.

7.2.2.- Glosario de conceptos.

- **Usuario:** Es cualquier persona que este registrada en el sistema. Un usuario tiene al menos un identificador, un mail, un nombre, un tipo y una contraseña. De él derivan el resto de los usuarios registrados.
- **Usuario cliente:** Un usuario cliente es aquel que utiliza las aplicaciones clientes (robot, portal web o móvil) para visualizar las actividades o servicios de las empresas y hoteles. El usuario cliente tiene unas preferencias. El usuario cuando se encuentre en un hotel registrado y lo desee puede unirse a ese hotel para recibir sus servicios.
- **Usuario hotel administrador:** Un hotel es una persona registrada en el sistema que representa a un hotel y puede ofrecer actividades y realizar notificaciones a sus huéspedes. El usuario hotel administrador puede tener

actividades y noticias. Además, este usuario puede registrar otros usuarios hotel, darles privilegios de hotel administrador y ver las estadísticas.

- **Usuario hotel:** Es un usuario que pertenece a un hotel determinado. Puede gestionar las actividades y noticias del hotel.
- **Usuario empresa administrador:** Un usuario empresa es una persona que representa a una empresa y ofrece actividades. La empresa gestiona actividades y visualiza estadísticas. Además, puede administrar los usuarios empresa que pertenecen a ella.
- **Usuario empresa:** Es un usuario que pertenece a una empresa determinada. Puede las gestionar actividades de la empresa.
- **Usuario administrador:** Es aquel que tiene la capacidad de gestionar todos los usuarios del sistema.
- **Usuario anónimo:** todo visitante que visualiza las actividades del sistema. No se registra de él ningún dato relevante.
- **Actividad o servicio:** Una actividad o servicio es toda aquella actividad que tiene una fecha de comienzo, de fin, duración, repetición, precio y otros datos relacionados.
- **Noticia:** Las noticias o notificaciones son textos que los hoteles pueden enviar a sus usuarios.
- **Categoría:** Se entiende como una forma de etiquetar la actividad.

7.2.3.- Modelo del diseño.

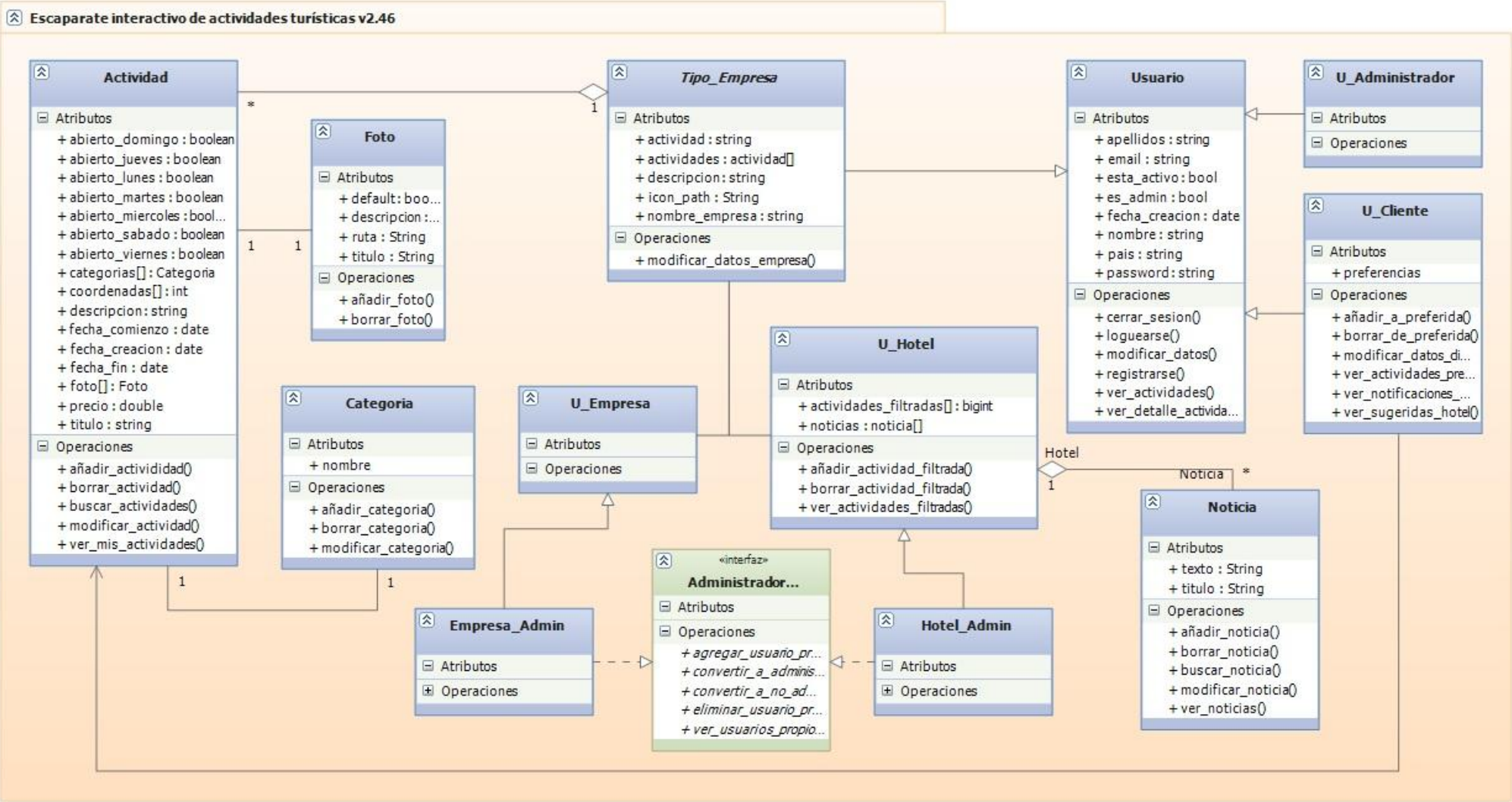


Figura. 7.2. Modelo de diseño

7.2.4.- Actores del software.

En el sistema habrá varios actores de software. A continuación se muestra un diagrama para facilitar la comprensión de las relaciones y un cuadro para describirlos en detalle.

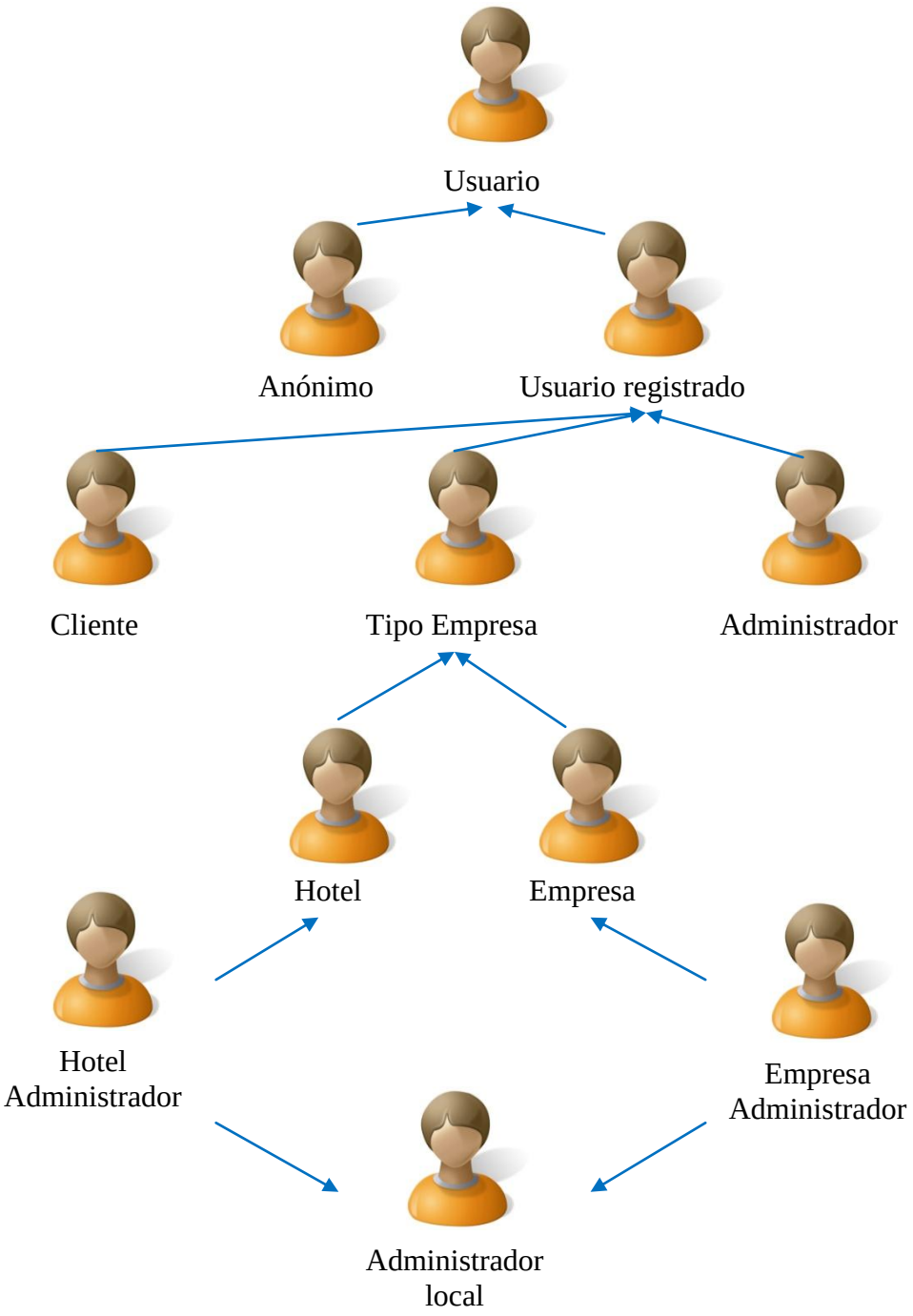


Figura. 7.3. Actores del software

Los perfiles efectivos del sistema son: **anónimo, cliente, administrador, hotel, hotel administrador, empresa y empresa administrador.**

7.2.5.- Listados de actores y sus roles.

Actor (rol)	Tipo	Definición
Usuario cliente	Principal	Un usuario cliente es aquel que utiliza las aplicaciones clientes (robot, portal web o móvil) para visualizar las actividades o servicios de las empresas y hoteles. El usuario cliente tiene unas preferencias. El usuario cuando se encuentre en un hotel registrado y lo desee puede unirse a ese hotel para recibir sus servicios.
Usuario hotel administrador	Principal	Es una persona registrada en el sistema que representa a un hotel real, y puede ofrecer actividades, realizar notificaciones a sus huéspedes y ver sus estadísticas. El usuario hotel puede tener actividades y notificaciones. Además, administra a los usuarios hotel de esa compañía.
Usuario hotel	Secundario	Es un usuario que pertenece a un hotel y gestiona las actividades y notificaciones del hotel.
Usuario empresa administrador	Principal	Una empresa es una persona que representa a una empresa. Gestiona actividades, usuarios de la empresa y visualiza las estadísticas.
Usuario empresa	Secundario	Es un usuario que pertenece a una empresa y que gestiona sus actividades.
Usuario administrador	Principal	Es aquel que tiene la capacidad de gestionar todos los usuarios.
Usuario anónimo	Principal	Todo visitante que visualiza las actividades que ofrece el sistema. No se registra de él ningún dato relevante. No obtiene información filtrada por hoteles porque no puede unirse a ellos. Puede registrarse y loguearse.

Tabla 7.1. Listado de actores y sus roles

7.2.6.- Listados de actores y sus objetivos.

Actor (rol)	Objetivos	Resumen de la acción
Usuario	Ver detalle actividad	Muestra en detalle la actividad seleccionada.
	Ver las top diez actividades	Se muestra una lista con las diez primeras actividades con mayor puntuación en visualizaciones
	Buscar actividades	Muestra una lista de las actividades en el sistema de acuerdo a los parámetros introducidos.
Usuario registrado	Cerrar sesión	Cierra la sesión actual del usuario para pasar a ser un usuario anónimo.
	Visualizar datos	Muestra los datos del usuario introducidos en el registro.
	Modificar datos	Guarda las modificaciones realizadas en los datos del usuario.
Usuario cliente	Ver preferencias	Obtiene un listado de las categorías de gustos que has seleccionado.
	Modificar preferencias	Guardas los cambios sobre el listado de preferencias mostrado.
	Ver actividades favoritas	Muestra una lista de las actividades favoritas añadidas anteriormente.
	Añadir a actividades favoritas	Añade una actividad a la lista de favoritas del usuario.
	Eliminar actividad favorita	Elimina la actividad seleccionada de la lista de favoritas del usuario.
	Ver actividad recomendada	Se muestra una actividad recomendada para el usuario basada en sus votaciones y preferencias
	Ver actividades sugeridas por el hotel	Muestra la lista de actividades recomendadas por el hotel al que se está vinculado.
	Ver notificaciones del hotel	Muestra una lista de las notificaciones que ha publicado el hotel al que se está unido.
	Ver actividades del hotel	Muestra una lista de las

		actividades que ofrece el propio hotel.
	Vincularse a hotel	Se vincula a un hotel para poder ver sus actividades sugeridas de otras empresas, las propias y sus notificaciones.
	Desvincularse del hotel	Te desvinculas del hotel anteriormente vinculado.
Tipo empresa	Añadir actividad	Añade una actividad a la lista de actividades del hotel.
	Ver mis actividades	Muestra las actividades del hotel.
	Borrar actividad	Borra la actividad seleccionada de la lista de actividades de la compañía.
	Modificar actividad	Modifica la actividad seleccionada de la lista de actividades de la compañía.
Usuario hotel	Añadir actividad filtrada	Estando en la lista de actividades, el usuario hotel añade una actividad a su lista de actividades filtradas.
	Añadir notificación	El usuario hotel añade una notificación a su lista de notificaciones.
	Borrar actividad filtrada	Borra la actividad seleccionada que se encuentra en la lista de actividades filtradas.
	Borrar notificación	Borra la noticia seleccionada de la lista de noticias.
	Modificar notificación	Guarda los cambios de la noticia seleccionada y visualizada.
	Ver actividades filtradas	Muestra la lista de actividades filtradas por el usuario hotel.
	Ver notificaciones	Muestra el listado de las noticias añadidas anteriormente.
Usuario administrador local	Ver estadísticas	Muestra las estadísticas de la empresa.
	Ver datos de empresa	Visualiza los datos de la empresa
	Modificar datos de empresa	Permite editar los datos de la empresa.
	Añadir usuario	Añade un usuario que pertenecerá a esa tipo empresa.

	Eliminar usuario	Elimina un usuario que pertenece a esa tipo empresa.
	Ver usuarios	Muestra todos los usuarios que pertenecen a esa tipo empresa.
	Modificar usuario	Modifica los datos del usuario seleccionado que pertenece a la empresa.
Usuario hotel administrador	(ninguno)	(ninguno en especial, sólo lo que hereda)
Usuario empresa	(ninguno)	(ninguno en especial, sólo lo que hereda)
Usuario empresa administrador	(ninguno)	(ninguno en especial, sólo lo que hereda)
Usuario administrador	Borrar usuario	Borra el usuario seleccionado.
	Añadir usuario	Muestra un formulario para añadir un usuario al sistema.
	Modificar usuario	Muestra el detalle del usuario seleccionado para modificarlo.
	Buscar usuario	Muestra una lista de usuarios que coinciden con los parámetros de búsqueda.
Usuario anónimo	Iniciar sesión	En el formulario de identificación se introducen los datos para quedar registrado en el sistema.
	Registrarse	En el formulario de registro permite al usuario anónimo quedar registrado en el sistema.

Tabla 7.2. Listados de actores y sus objetivos

7.2.7.- Diagramas UML de casos de uso.

Usuario

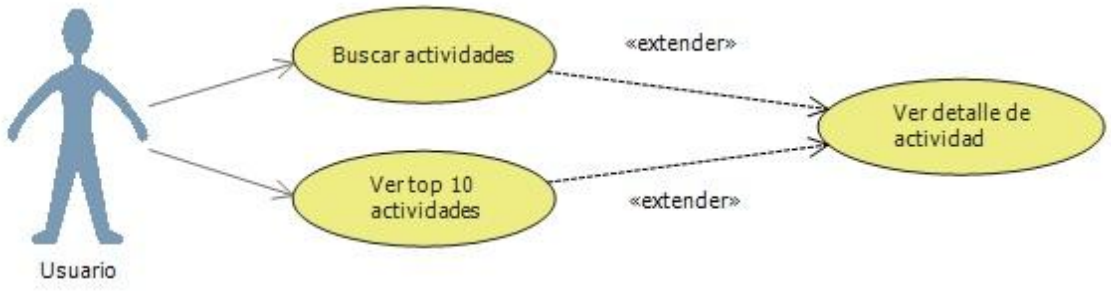


Figura. 7.4. Caso de uso del usuario

Usuario registrado

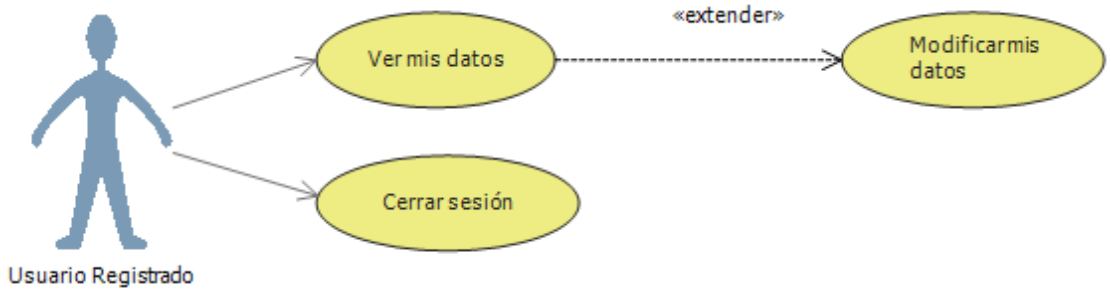


Figura. 7.5. Caso de uso del usuario registrado

Usuario tipo empresa

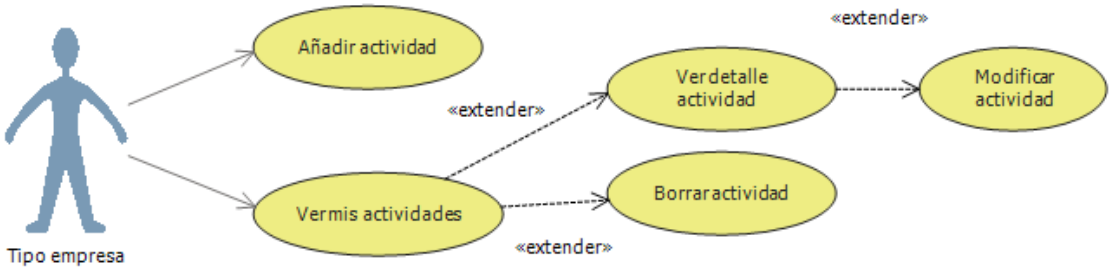


Figura. 7.6. Caso de uso del usuario tipo empresa

Usuario Hotel

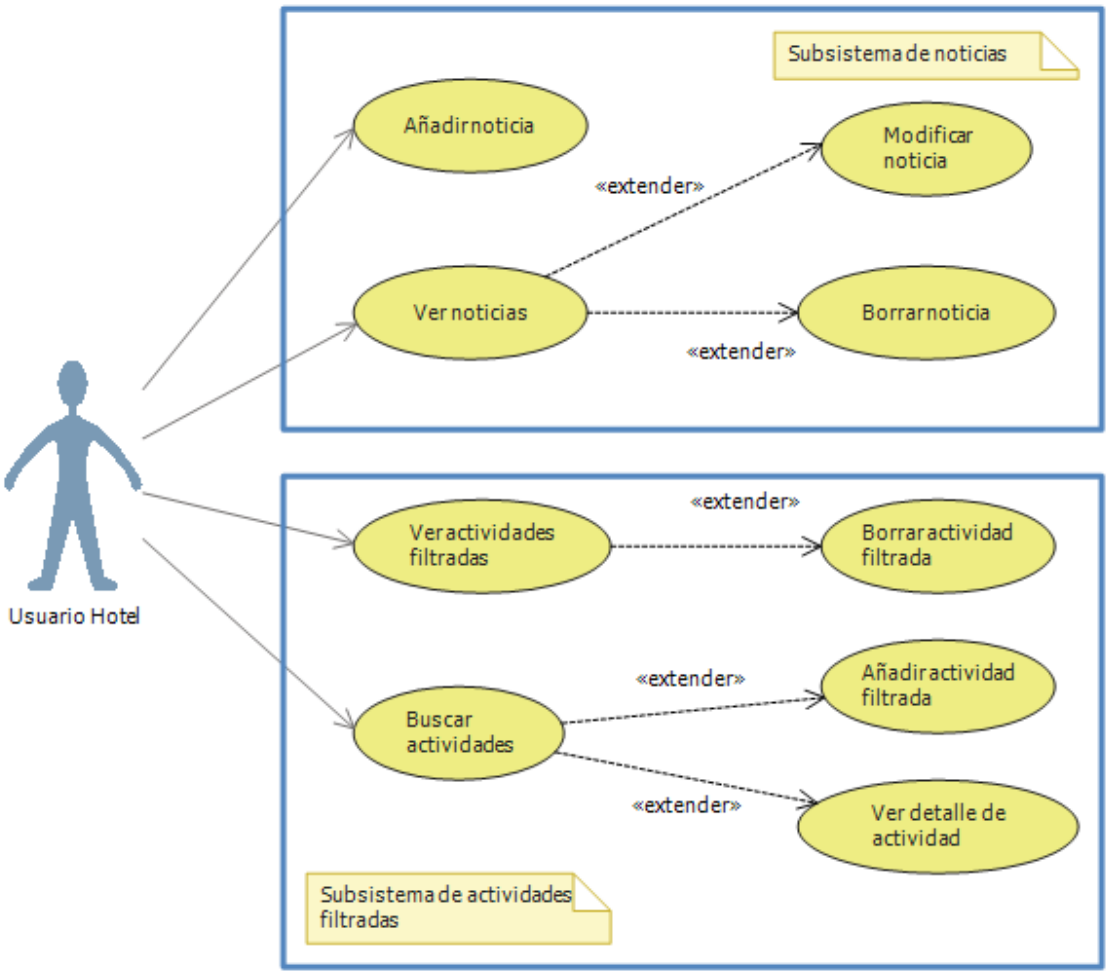


Figura. 7.7. Caso de uso del usuario hotel

Usuario Administrador Local

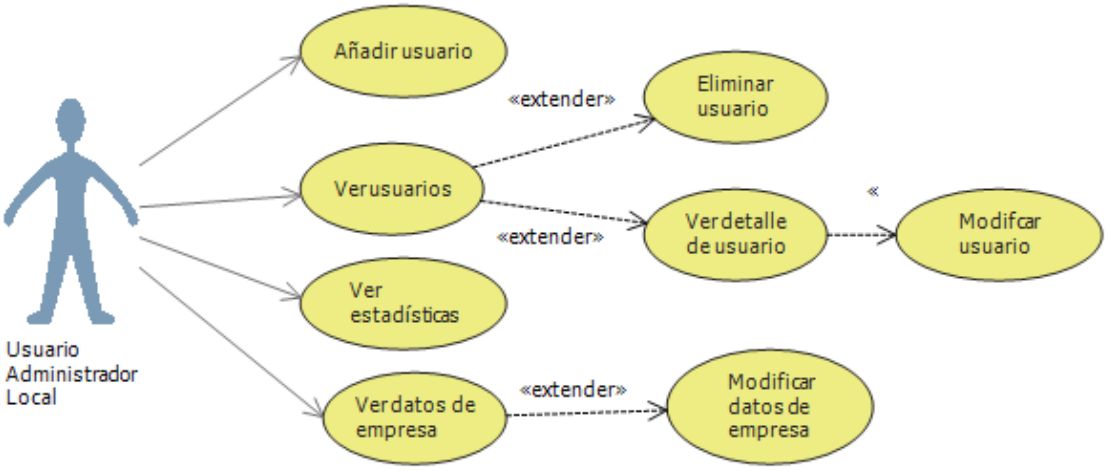


Figura. 7.8. Caso de uso usuario administrador local

Usuario Administrador

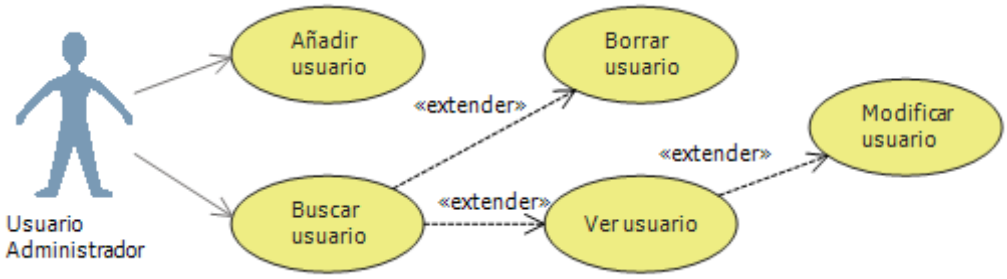


Figura. 7.9. Caso de uso usuario administrador

Usuario Anónimo

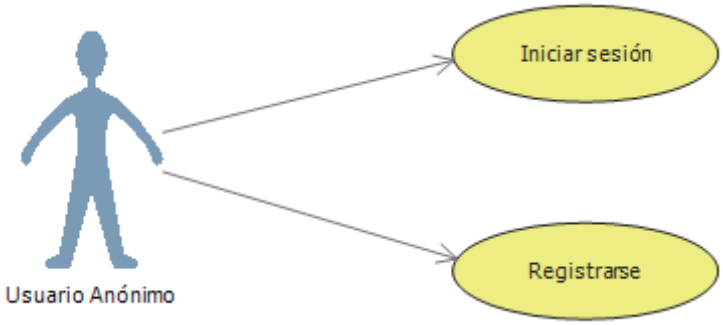


Figura. 7.10. Caso de uso usuario anónimo

Usuario Cliente

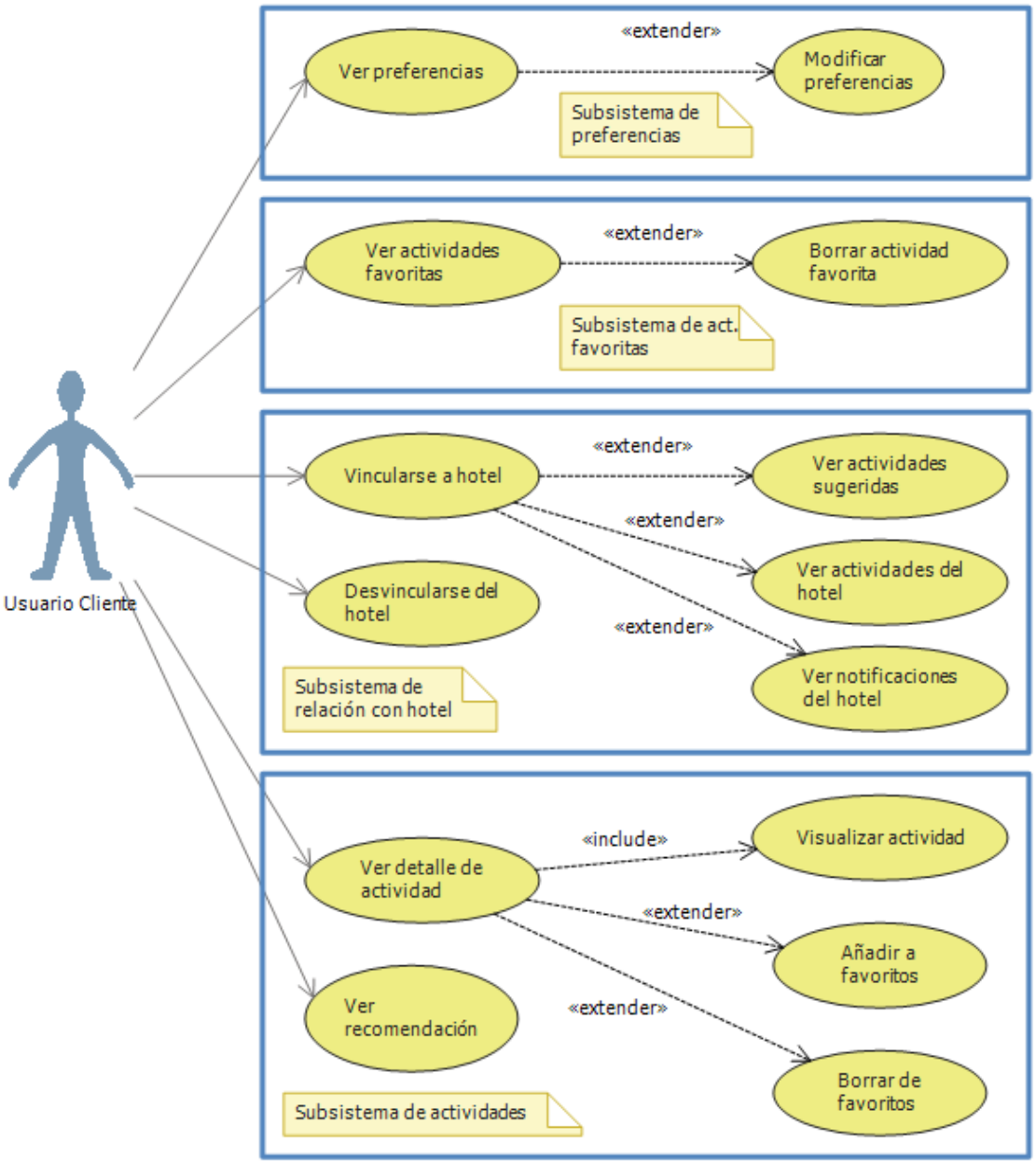


Figura. 7.11. Caso de uso del usuario cliente

7.2.8.- Listado de casos de uso.

Actor (rol)	Caso de uso	Nº CU
Usuario	Ver detalle actividad	01
	Ver las top diez actividades	02
	Buscar actividades	03
Usuario registrado	Cerrar sesión	04
	Visualizar datos	05
	Modificar datos	06
Usuario cliente	Ver preferencias	07
	Modificar preferencias	08
	Ver actividades favoritas	09
	Añadir a actividades favoritas	10
	Eliminar actividad favorita	11
	Ver actividad recomendada	12
	Ver actividades sugeridas por el hotel	13
	Ver notificaciones del hotel	14
	Ver actividades del hotel	15
	Vincularse a hotel	16
	Desvincularse del hotel	17
Tipo empresa	Añadir actividad	18
	Ver mis actividades	19
	Borrar actividad	20
	Modificar actividad	21
Usuario hotel	Añadir actividad filtrada	22
	Añadir notificación	23
	Borrar actividad filtrada	24
	Borrar notificación	25
	Modificar notificación	26
	Ver actividades filtradas	27
	Ver notificaciones	28
Usuario administrador local	Añadir usuario	29
	Eliminar usuario	30
	Modificar usuario	31
	Ver usuarios	32
	Ver estadísticas	33
	Ver datos de empresa	34
	Modificar datos de empresa	35
Usuario administrador	Borrar usuario	36
	Añadir usuario	37
	Modificar usuario	38
	Buscar usuario	39
Usuario anónimo	Iniciar sesión	40

	Registrarse	41

Tabla 7.3. Listado de casos de uso

7.2.9.- Diagramas de secuencia.

Los diagramas de secuencia están elaborados en lenguaje UML y representan de forma esquemática las acciones que debe ir realizando un rol en el sistema a lo largo de un periodo de tiempo para completar una acción. La elaboración de estos diagramas se apoya en el desarrollo previo de los diagramas de clases y los casos de uso que son elementos típicos en la ingeniería del software.

Ver detalle de una actividad.

La siguiente secuencia representa uno de los fines del sistema que es ofrecer datos de las actividades a los usuarios. En este diagrama están incluidas otras secuencias como buscar actividades:

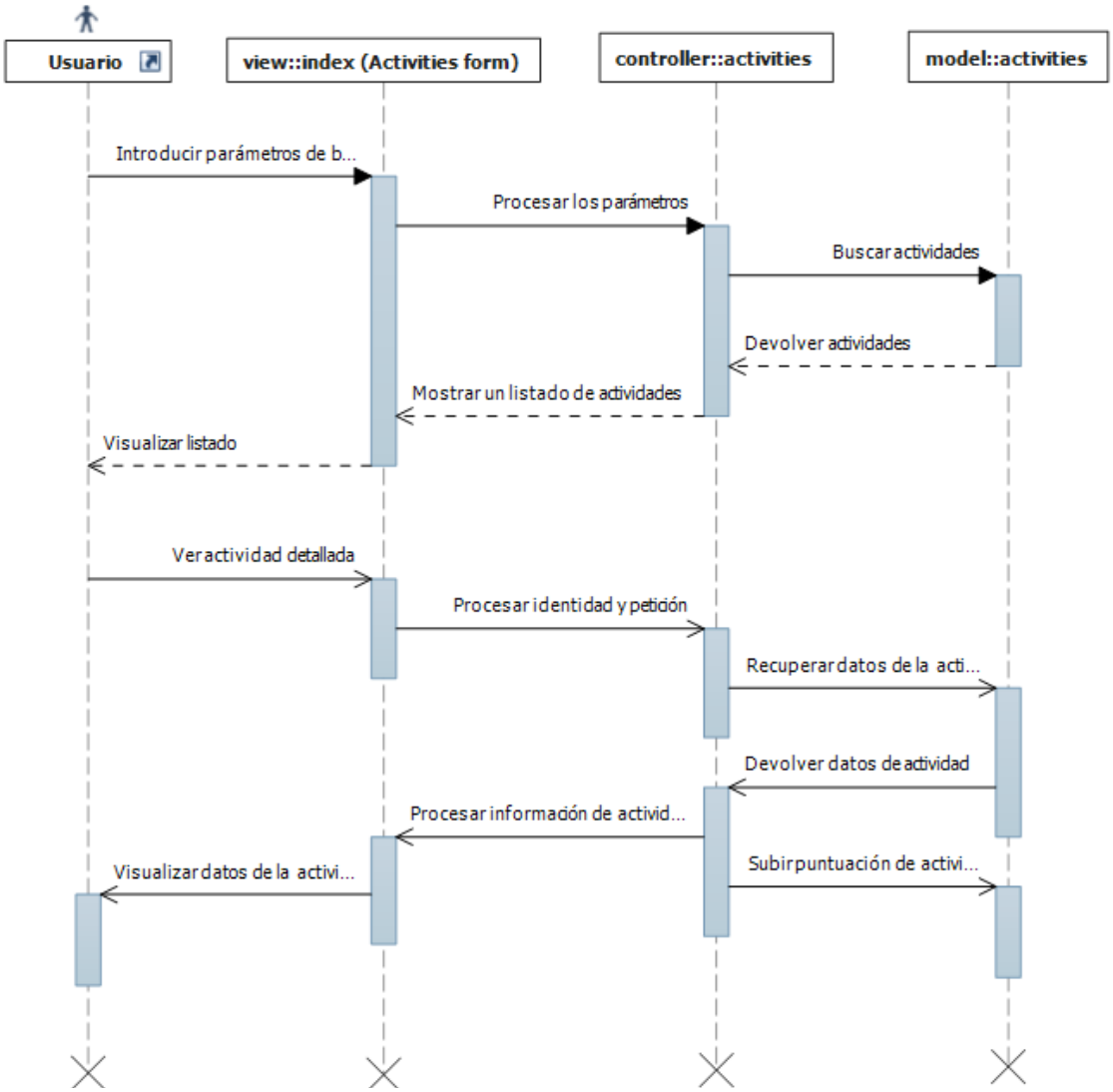


Figura. 7.12. Diagrama de secuencia de visualización de actividad

El actor principal de este diagrama es un usuario del sistema, ya se encuentre registrado o no. Este usuario rellenará un formulario con los parámetros de

búsqueda acordes al tipo de actividad que desea. El controlador recibirá la petición y procesará los datos del formulario para que o bien sean corregidos o se genere una petición a la base de datos. La base de datos buscará todas las actividades que se encuentren dentro del rango de búsqueda generado por el controlador y devolverá una lista. El controlador procesará la lista para generar una vista paginada para el usuario.

Modificar datos de usuario.

Esta acción comprende de nuevo otras acciones del sistema y resulta genérica y extensible a otros diagramas de secuencia como modificar preferencias.

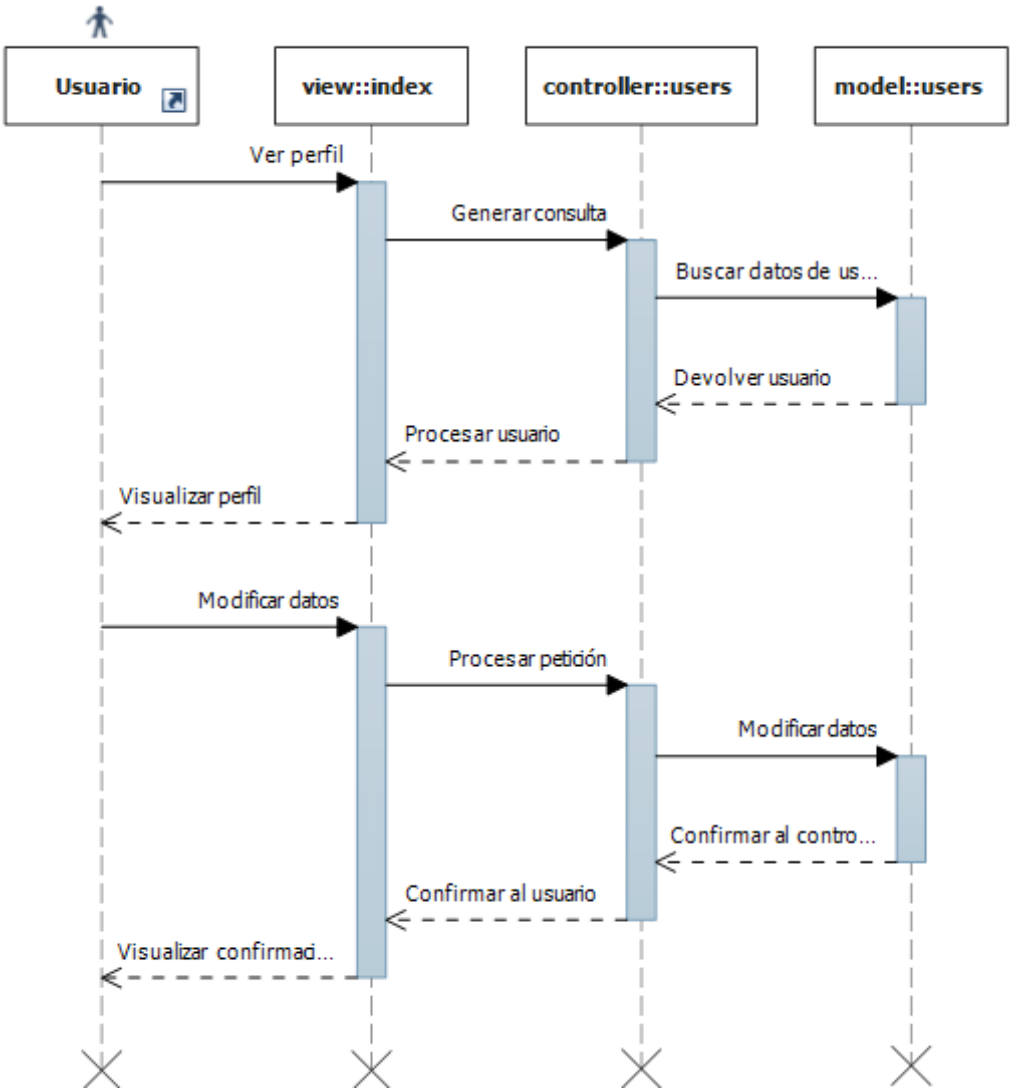


Figura. 7.13. Diagrama de secuencia de modificar datos de usuario

Para este diagrama de secuencia el actor principal es un usuario registrado en el sistema, por lo que previamente ha introducido sus datos en el mismo.

El usuario visualiza su perfil realizando una petición al controlador y este devolverá sus datos. Después de que el usuario haga las modificaciones pertinentes y pulse sobre modificar datos, el controlador procesará los nuevos datos introducidos en el sistema y si son validos los enviará al modelo para que sean sustituidos en la base de datos.

Añadir actividad.

Finalmente, la secuencia de añadir actividad resulta análoga a otras secuencias como eliminar actividad y modificar actividad.

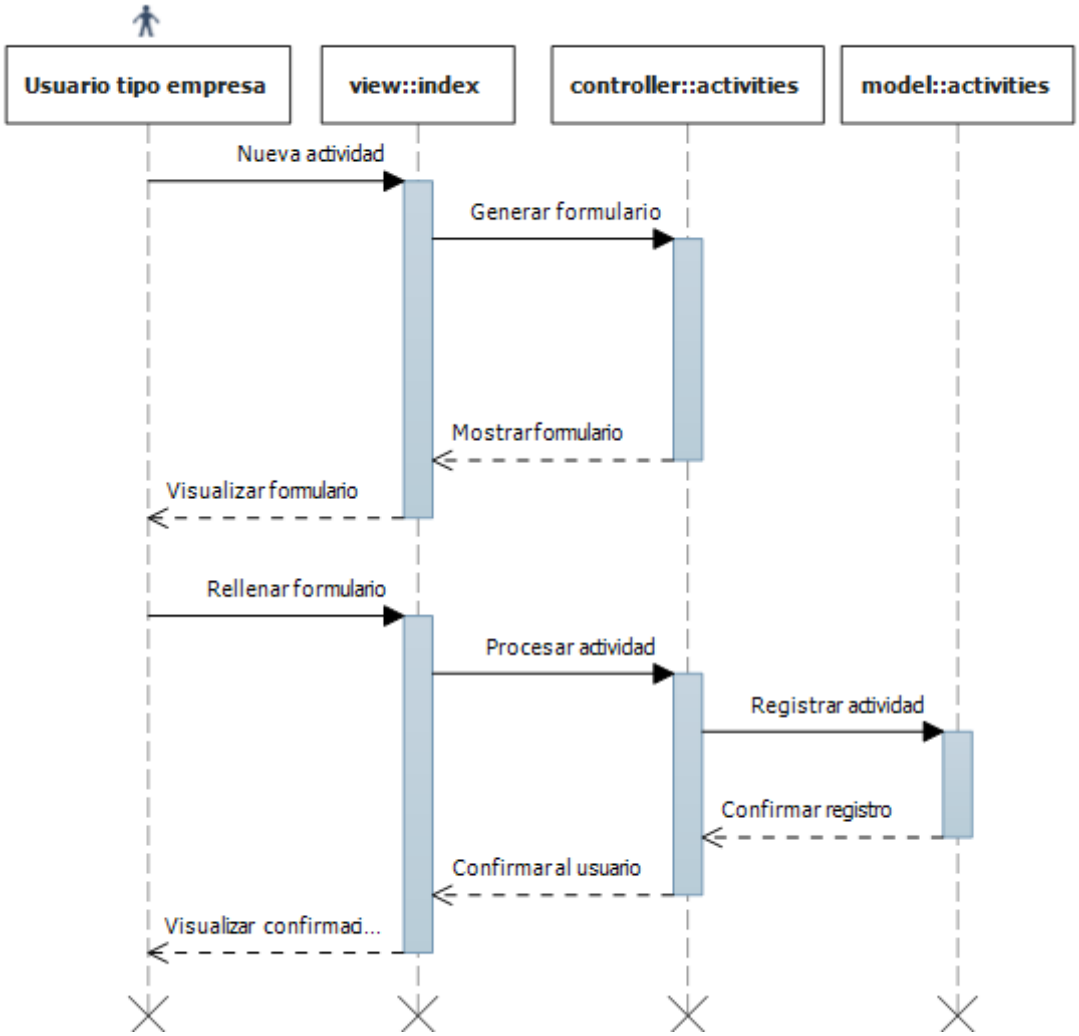


Figura. 7.14. Diagrama de añadir actividad

El usuario tipo empresa envía una petición para registrar una nueva actividad al controlador. Este la genera y el usuario envía el formulario de nuevo al controlador para que se procese. Posteriormente el modelo los almacena.

Unirse a un hotel.

Unirse al hotel es una de las tareas que puede realizar un usuario cliente para recibir notificaciones, visualizar las actividades del propio hotel y las que recomienda. El siguiente diagrama de secuencia muestra cómo se produce este proceso de unión.

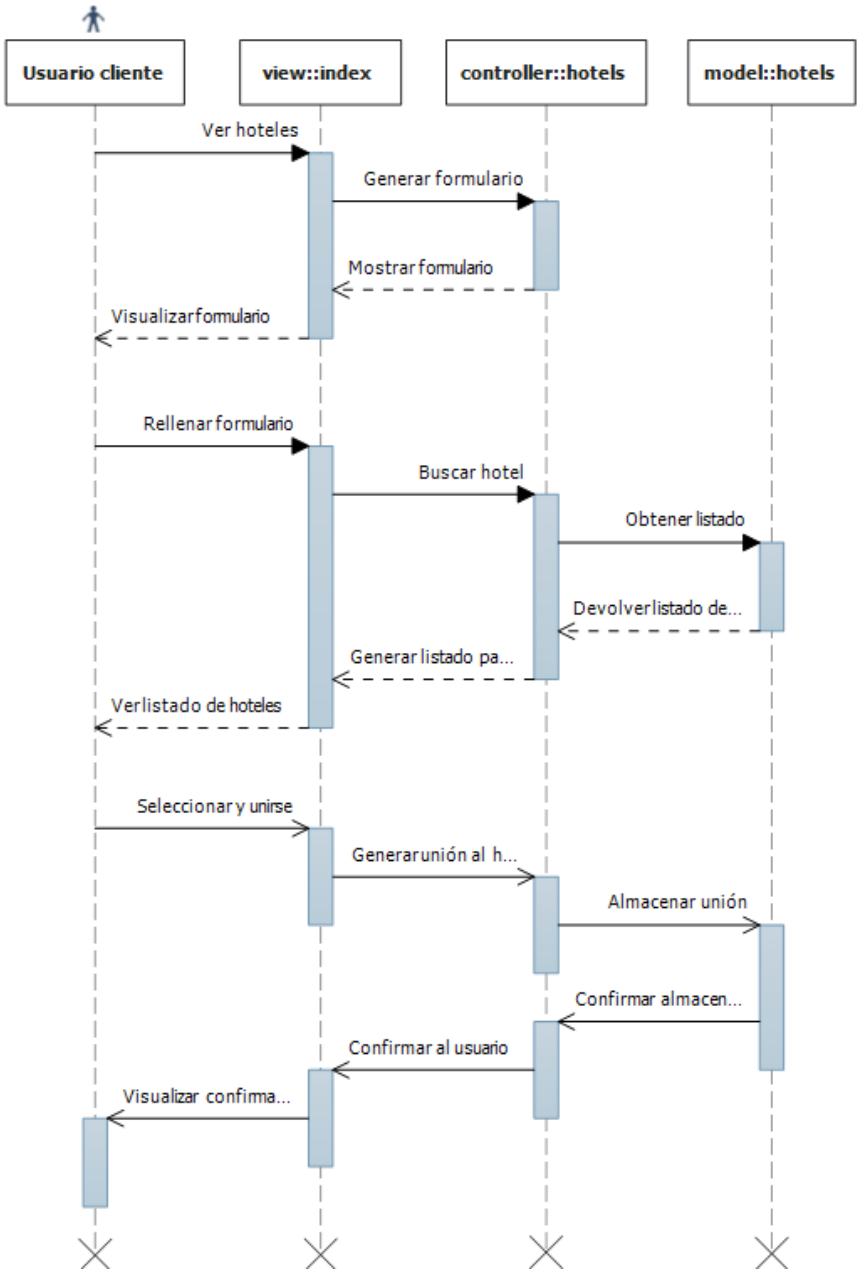


Figura. 7.15. Diagrama de secuencia de unirse a un hotel

Para poder unirse al hotel es necesario tener el rol de usuario cliente en el sistema.

El usuario cliente debe primeramente buscar por nombre el hotel al que quiere asociarse y controlador generará una consulta de los hoteles que concuerden con

el nombre introducido. A continuación el controlador recibirá del modelo un listado de los hoteles que le mostrará al usuario de forma paginada. El usuario seleccionará un hotel y se unirá. El controlador procesará la petición e indicará al modelo que almacene esa unión en el sistema.

7.2.10.- Prototipos de las interfaces de usuario.

Los prototipos de las interfaces de usuario son unos elementos muy importantes en la ingeniería del software y ayudan a representar el aspecto final que debe tener la aplicación.

Interfaz principal de usuario anónimo.

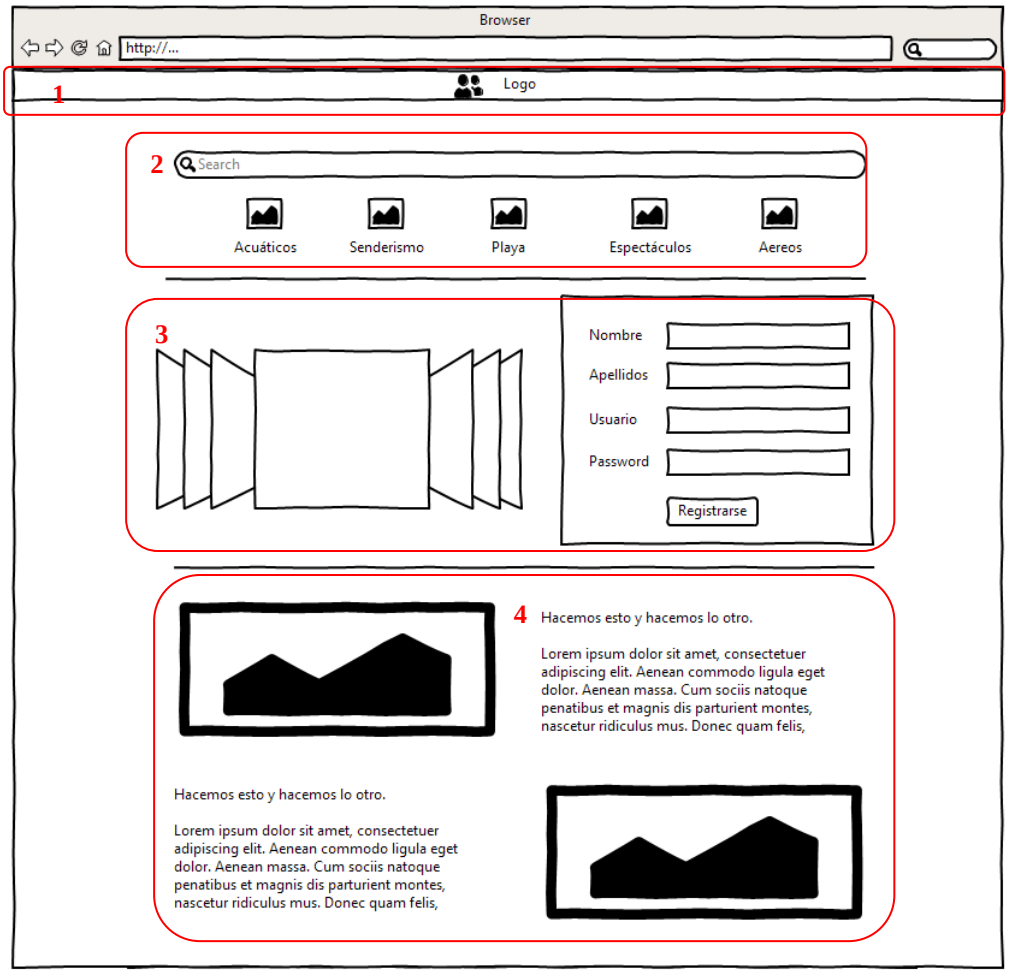


Figura. 7.16. Prototipo de interfaz principal del usuario anónimo

En la interfaz de se diferencian varios módulos. En la parte superior de la página (1) se encuentra una barra que cobra toda su utilidad cuando el usuario ha iniciado sesión. A continuación (2) se puede distinguir un formulario que actúa como buscador de actividades en el sistema y permite introducir varios parámetros para personalizar aún más las búsquedas.

Inmediatamente debajo (3) aparecen otros dos módulos, a la izquierda un rotatorio de actividades sugeridas y a la derecha un formulario para registrarse en el sistema.

Continuando hacia abajo (4), se distribuyen varias imágenes y textos que definen el propósito, las características y funciones de la aplicación.

Interfaz principal de usuario cliente.

Figura. 7.17. Prototipo de interfaz principal del usuario cliente

La interfaz principal del usuario cliente hereda un módulo que heredarán el resto de interfaces en casi todos los casos, el módulo de búsqueda de actividades. Este módulo estará presente en casi todo momento en la aplicación ya que el objetivo principal es que los clientes tengan siempre a mano las actividades que desean.

En esta interfaz personalizada para el usuario cliente, el módulo uno mostrará una actividad sugerida en función del perfil y las preferencias del cliente. El usuario tendrá visibles las opciones de incluirlo a favorito y votarlo.

En el módulo dos se encuentra el detalle de la actividad sugerida, que incluye los horarios, el tipo de actividad y un mapa para localizarla.

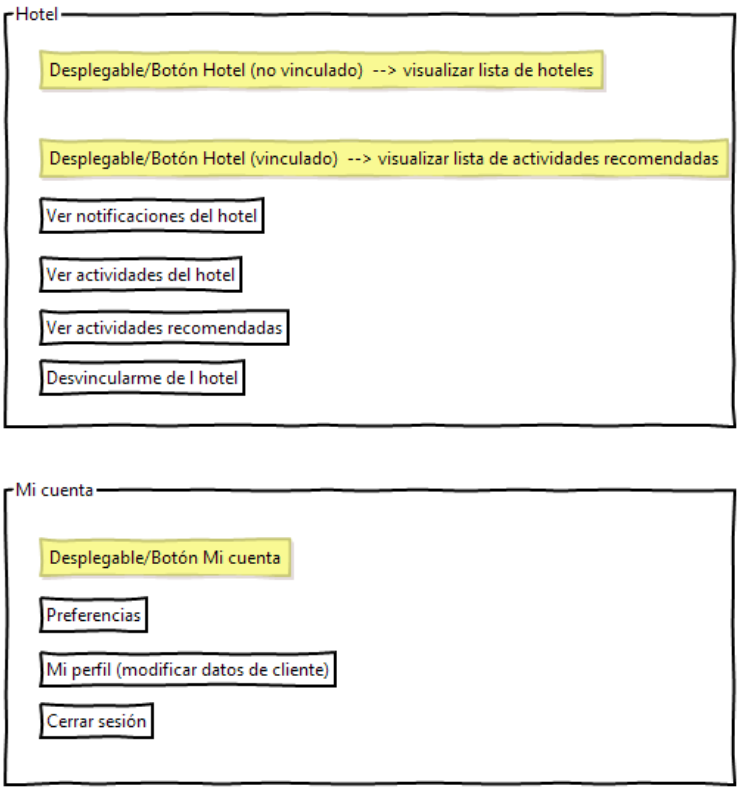


Figura. 7.18. Prototipo de interfaz del menú de usuario cliente

Los botones *hotel* y *mi cuenta* son desplegables. El botón hotel, cuando el usuario no está a un hotel, redirige al usuario a una página donde buscar hoteles y poder unirse. Por el contrario, cuando el usuario se encuentra unido a un hotel aparece un desplegable donde puede acceder a las notificaciones, actividades propias y de terceros sugeridas por el hotel.

En el desplegable *mi cuenta* están ubicados los accesos a las preferencias del usuario, el perfil y el cierre de sesión.

Interfaz de actividades de usuario empresa.

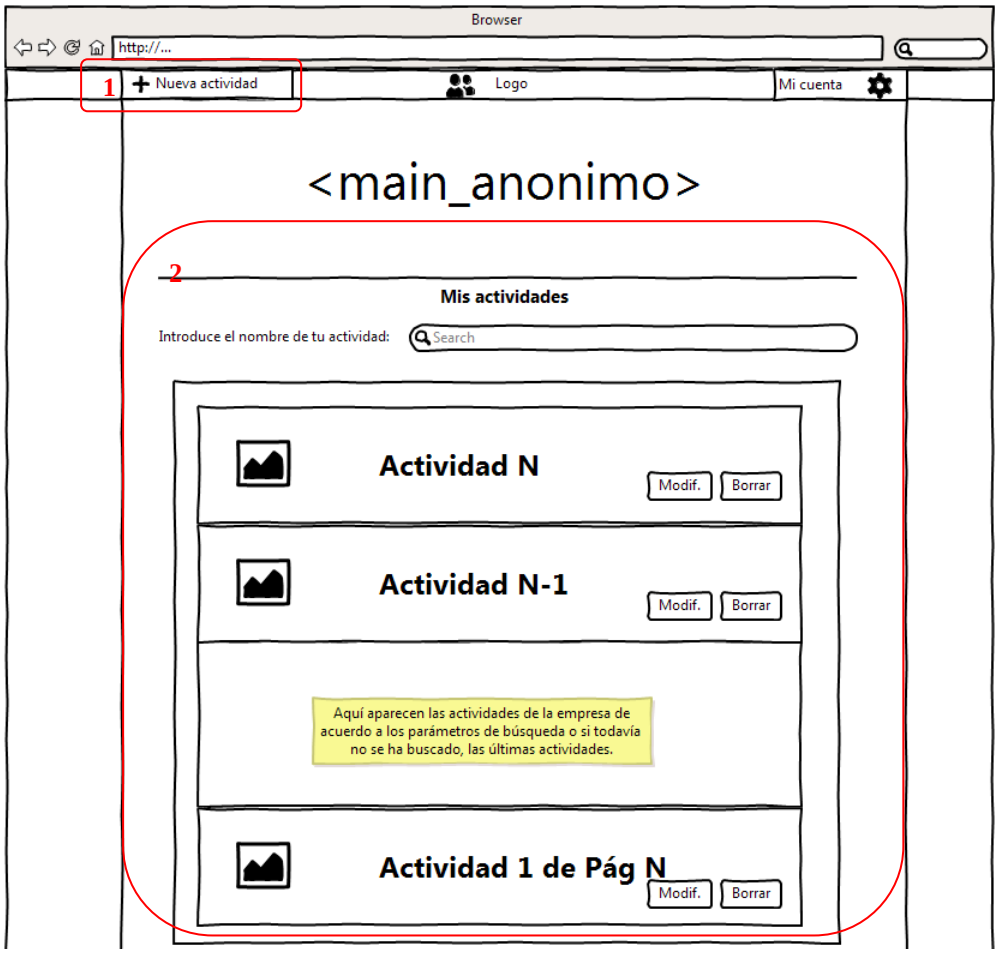


Figura. 7.19. Prototipo de interfaz de actividades del usuario empresa

El cometido principal de una empresa en la aplicación es tener actividades que los usuarios cliente quieran ver. Podemos observar como el usuario empresa podrá visualizar sus actividades (2). Desde esa interfaz es posible modificarlas o borrarlas. Para crear una nueva habrá siempre visible en la barra superior (1) el botón *nueva actividad*.

Interfaz principal de usuario hotel.

Figura. 7.20. Prototipo de interfaz principal del usuario hotel

El usuario hotel cuenta con un pequeño panel en su interfaz. Este panel incluye los accesos a las funciones que puede desempeñar en el sistema. Estos accesos corresponden a estadísticas, mensajes, actividades propias y recomendaciones.

Las estadísticas se muestran como gráficos a partir de los datos obtenidos de las interacciones de los usuarios clientes con ese hotel.

El módulo de mensajes comprende la gestión de mensajería que los usuarios pueden recibir.

Para las actividades existe un módulo análogo al que disponen los usuarios empresa en su interfaz.

Y finalmente las recomendaciones presentan una interfaz muy similar al de los anteriores, siendo un listado con opciones de gestión.

Interfaz específica de usuario cliente. Listar buscar actividades.

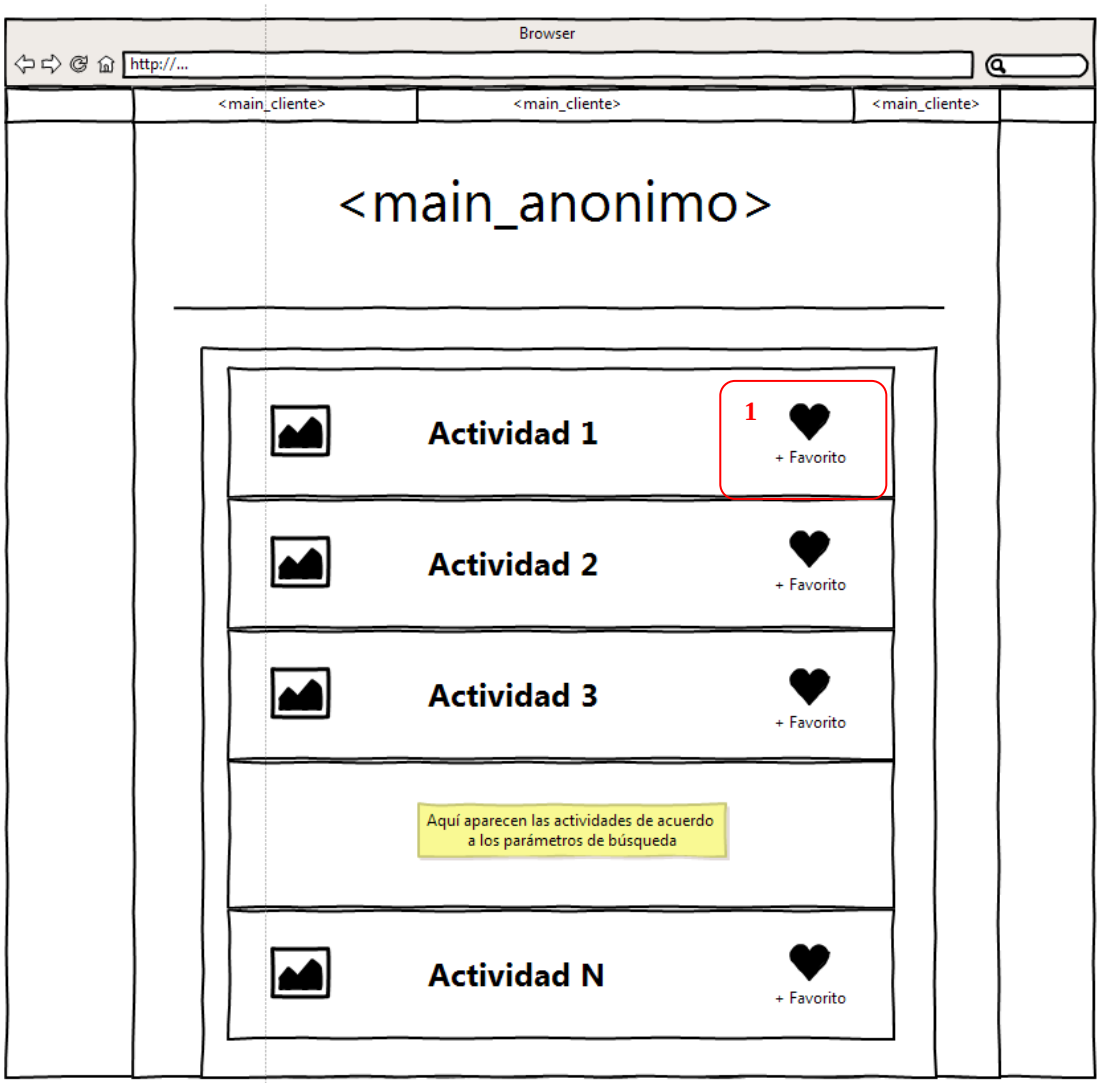


Figura. 7.21. Prototipo de interfaz principal del usuario cliente buscando o listando actividades

Cuando un usuario (general) busca actividades se muestra una interfaz muy parecida a la anterior. La intención es que el usuario cliente tenga además la opción (1) de indicar que quiere seleccionar una actividad de la lista como favorita o eliminarla de su lista si ya la tuviera.

En la lista de actividades se muestra la foto, el título y una porción de la descripción, dejando como opción del visitante ver los datos detallados de una actividad. Esto permite tener una interfaz limpia en la que no se muestren a priori todos los datos y mapas de todas las actividades listadas sino que se muestren dinámicamente a petición del usuario.

Interfaz específica de usuario hotel. Listar buscar actividades.

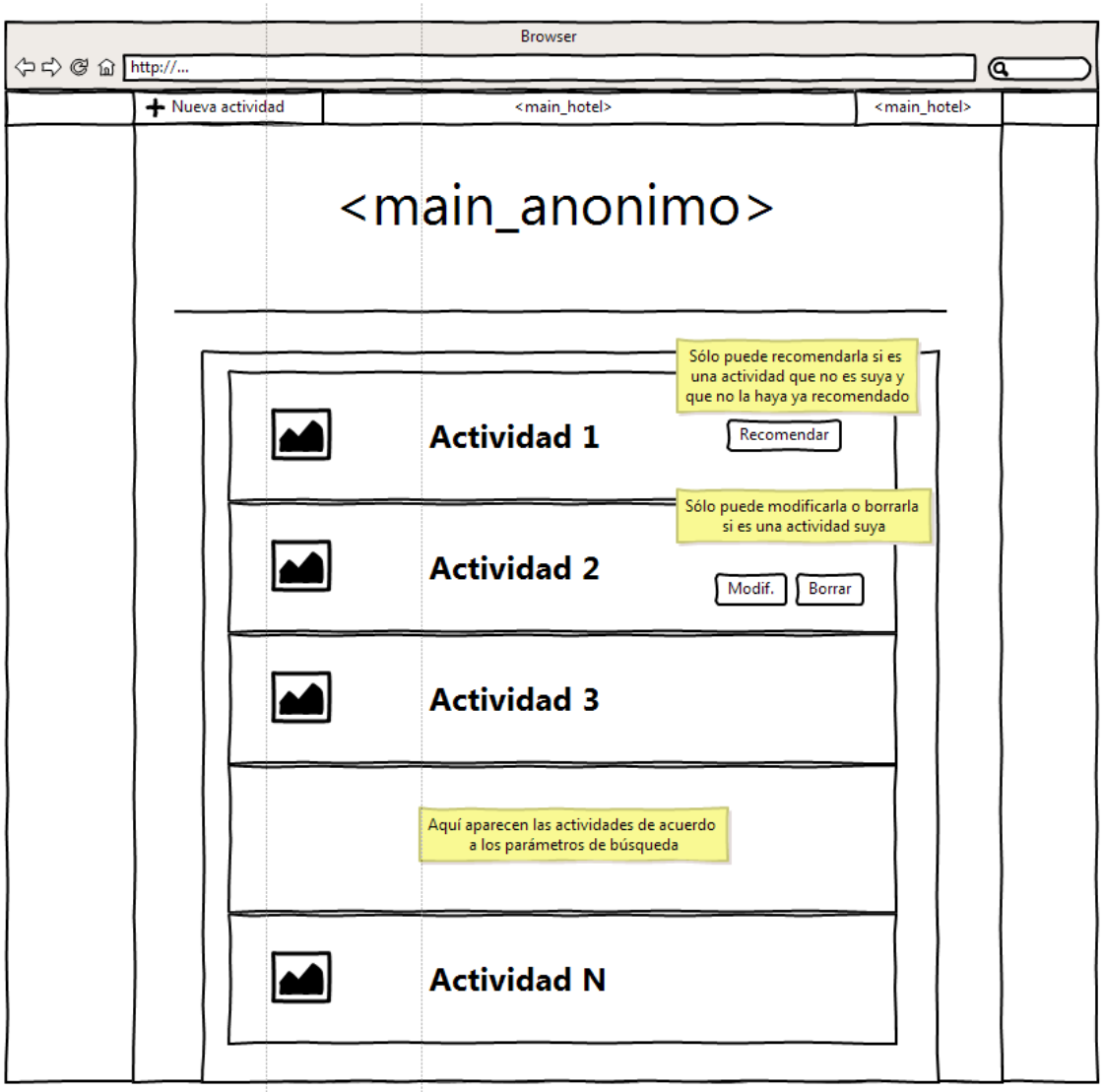


Figura. 7.22. Prototipo de interfaz para el usuario hotel listando o buscando actividades

Las interfaces específicas de los usuarios hotel y empresa cuando listan y buscan actividades son prácticamente iguales a diferencia del botón *recomendar*.

De nuevo, el objetivo principal de este vista es generar una apariencia minimalista con la que el gestor de actividades, ya sea una empresa o un hotel, se sienta cómodo para las funciones que puede desempeñar.

El sistema muestra un sencillo listado de actividades con el que el usuario hotel puede interactuar. En caso de que una actividad sea anunciada por el mismo le aparecerán las opciones de modificarla y borrarla. Si por el contrario es una actividad de terceros, podrá recomendarla en su hotel.

Interfaz específica de usuario hotel. Listar notificaciones.

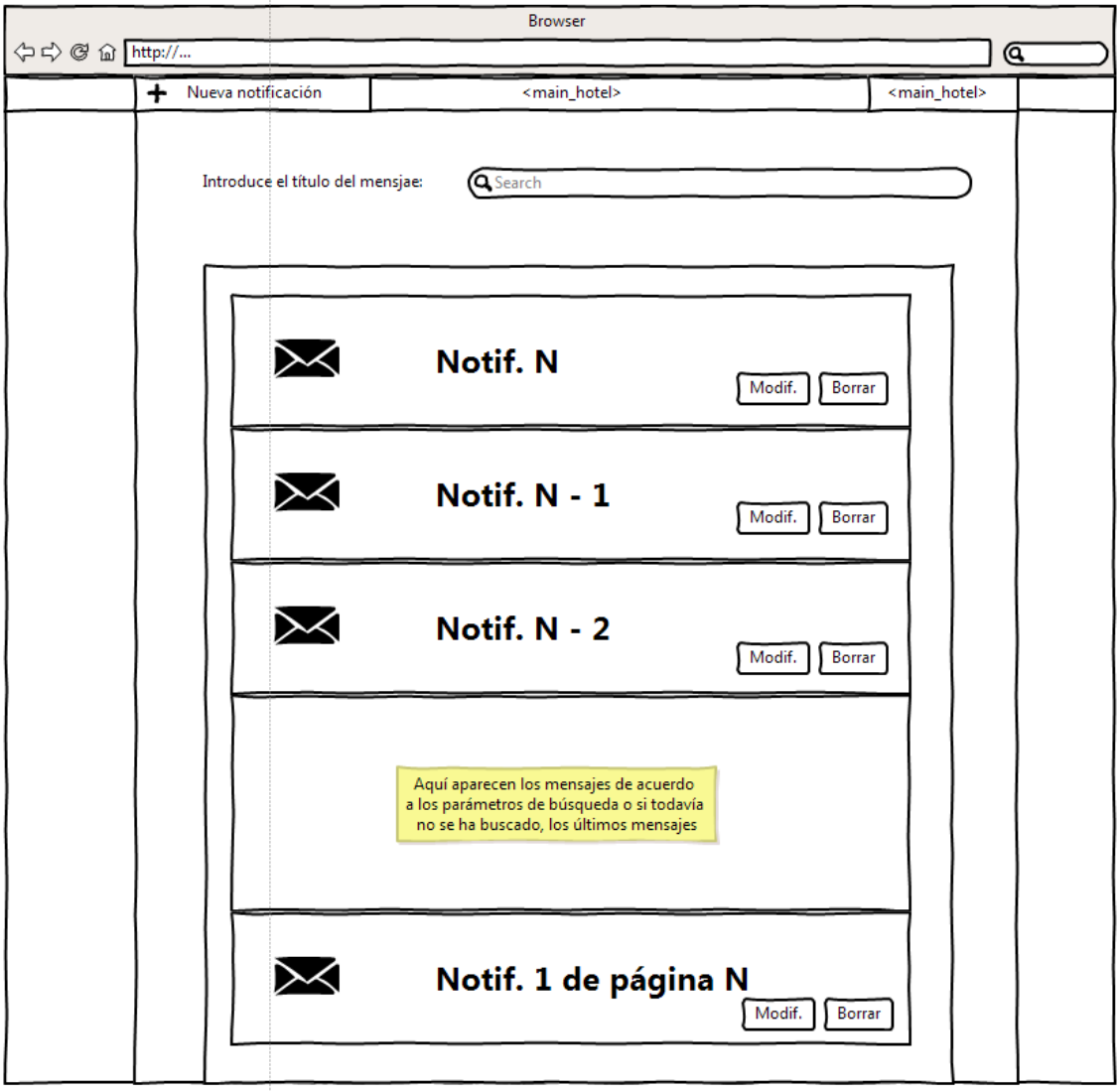


Figura. 7.23. Prototipo de interfaz de usuario del usuario hotel listando notificaciones

Tras traspasar el panel de usuario hotel y dirigirse al módulo de notificaciones debe aparecer algo similar a la figura anterior.

El usuario debe encontrarse cómodo y accesibles los botones. Con el objetivo de cumplir esto la interfaz del módulo de notificaciones debe tener una estructura similar al de actividades y otras gestiones en la aplicación.

Como las notificaciones son textos simples, se pueden listar de la misma manera que las actividades, mostrando el título de la notificación y sólo una parte del texto.

La vista agrupa las funciones de modificación, borrado y creación de notificaciones, para su gestión básica en una única interfaz y sin aglomeraciones de elementos innecesarios.

Interfaz específica de usuario hotel. Crear notificación.

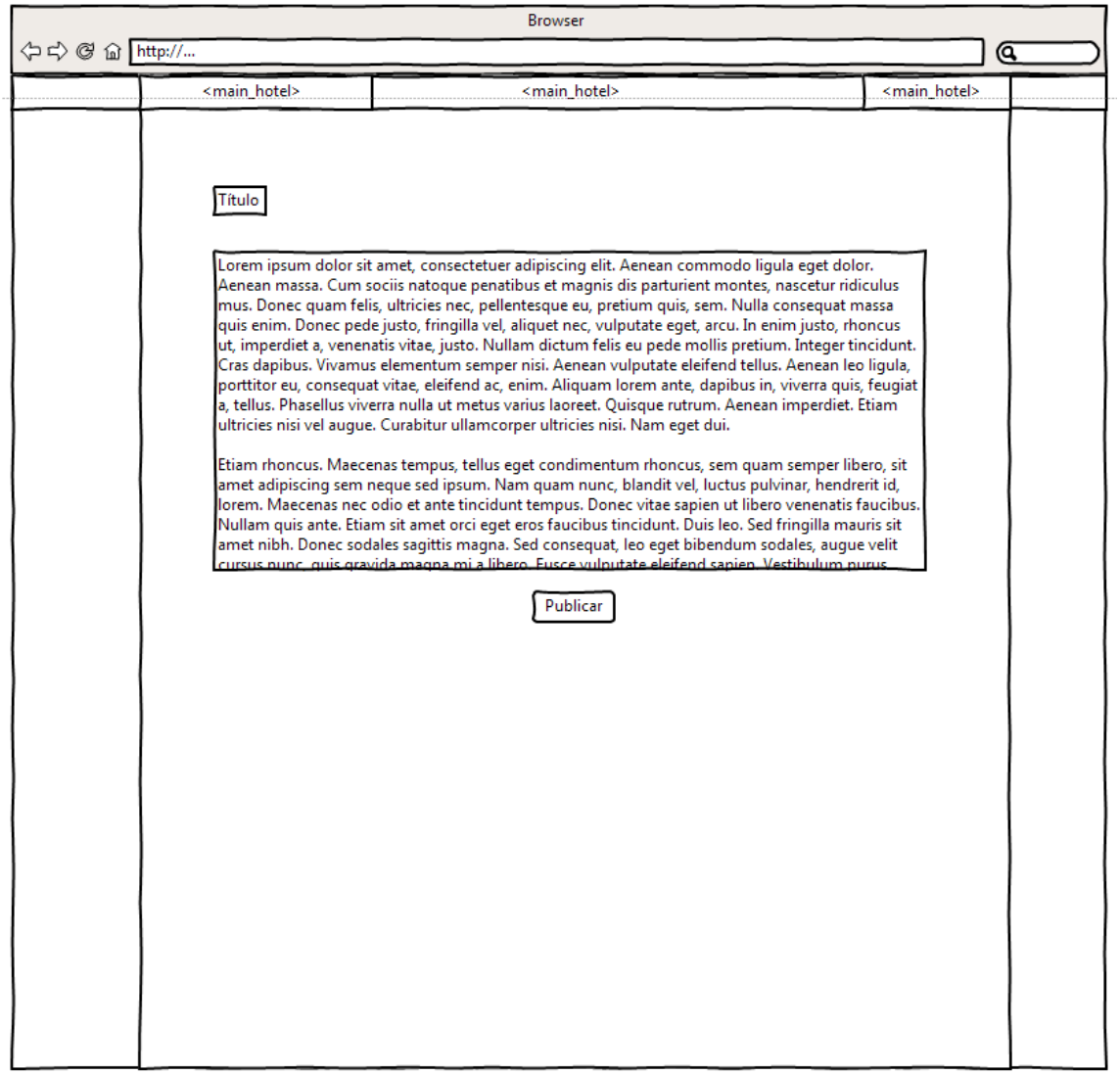


Figura. 7.24. Prototipo de interfaz de usuario hotel creando una nueva notificación

El formulario de creación de notificaciones debe tener únicamente dos campos, el título y el texto. Siguiendo el estilo de las interfaces anteriores, esta vista sólo muestra estos dos elementos, pero debe mostrarlos lo suficientemente amplios para que el gestor de notificaciones se encuentre cómodo en la interfaz.

7.2.11.- Base de datos.

Introducción.

El esquema de una base de datos describe la estructura de una base de datos. Este lenguaje es relacional y permite definir cada tabla con sus campos en el sistema y las relaciones entre ellas.

Las bases de datos permiten que los sistemas sean dinámicos y son herramientas potentes y ampliamente utilizadas en todas las aplicaciones.

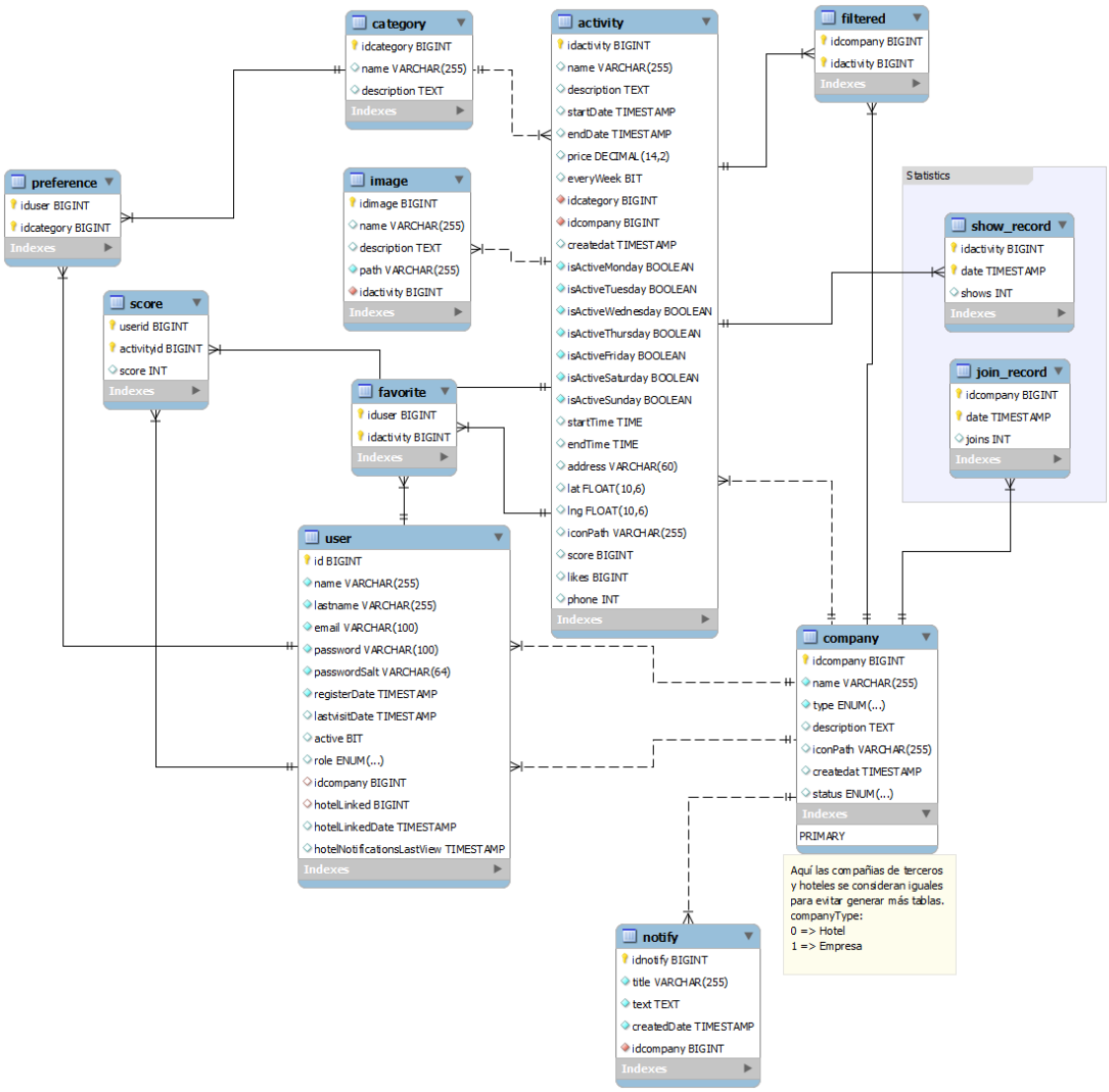


Figura. 7.25. Esquema de la base de datos

En el esquema anterior de la base de datos de la aplicación se distinguen doce tablas de datos. Estas tablas guardan los datos de los objetos de la aplicación descritos a lo largo de esta memoria.

La función general de cada tabla es la siguiente:

- Activity: Almacena cada una de las actividades que se registran en el sistema, ya sean de hoteles o de otras empresas.
- Category: Contiene una lista de tipos de actividades de la aplicación.
- User: Mantiene todos los usuarios registrados.
- Company: Guarda los datos de las compañías.
- Favorite: Almacena las actividades favoritas de los usuarios
- Notify: Contiene todas las notificaciones de las compañías.
- Filtered: Es la tabla donde se encuentra todas las actividades filtradas de los hoteles.
- Preference: Guarda las preferencias de las categorías de los usuarios clientes.
- Score: Mantiene la puntuación que ha dado cada usuario a cada actividad.
- Show_record: Almacena las visualizaciones de las actividades.
- Join_record: Guarda las uniones con cada uno de los hoteles.

Interrelaciones entre las tablas.

En las bases de datos es posible definir relaciones entre las tablas. Esto permite asemejarse más al modelo real y crear restricciones cuando se modifican.

Activity-filtered-company.

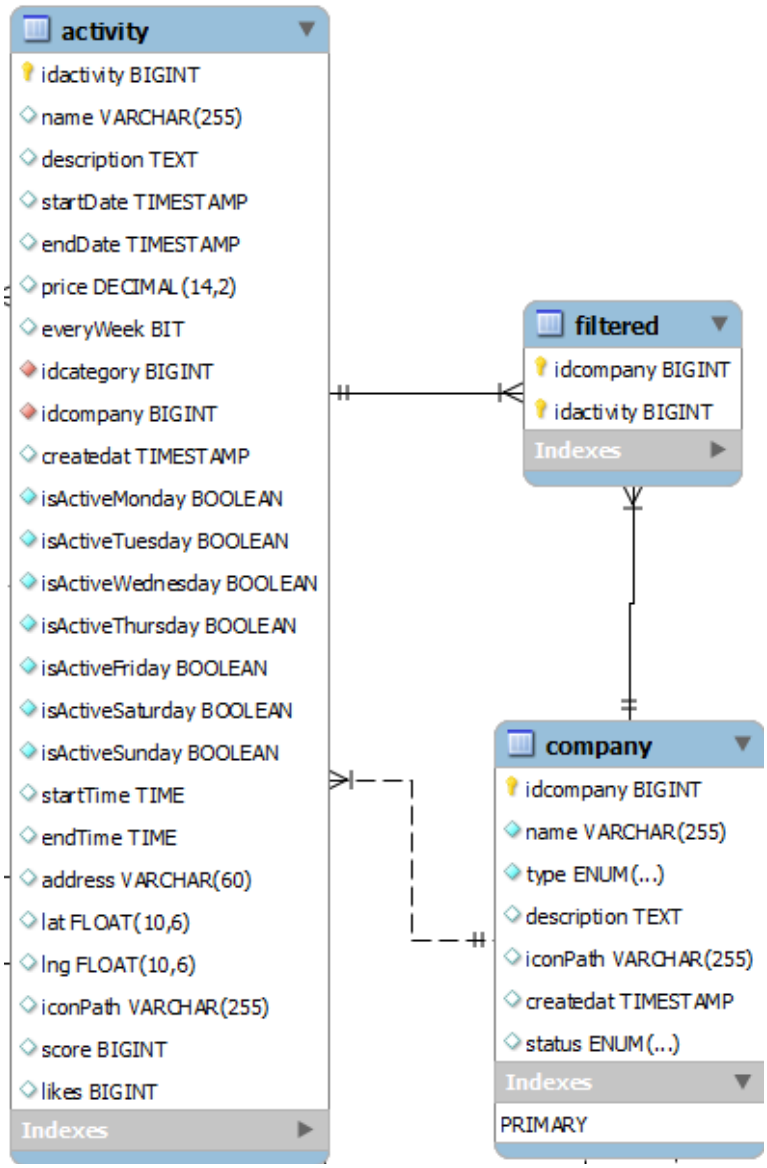


Figura. 7.26. Base de datos. Interrelación activity-filtered-company

Se pueden observar tres relaciones entres las tres tablas.

La relación entre las tablas *activity* y *company* representa una clave foránea. Cada una de las actividades pertenece a una compañía concreta. Una compañía puede tener muchas actividades pero una actividad sólo puede tener una

compañía asociada. Es una relación de uno a muchos. En la tabla *activity* existe un campo idcompany que referencia a la compañía que pertenece la actividad.

Activity-favorite-user.

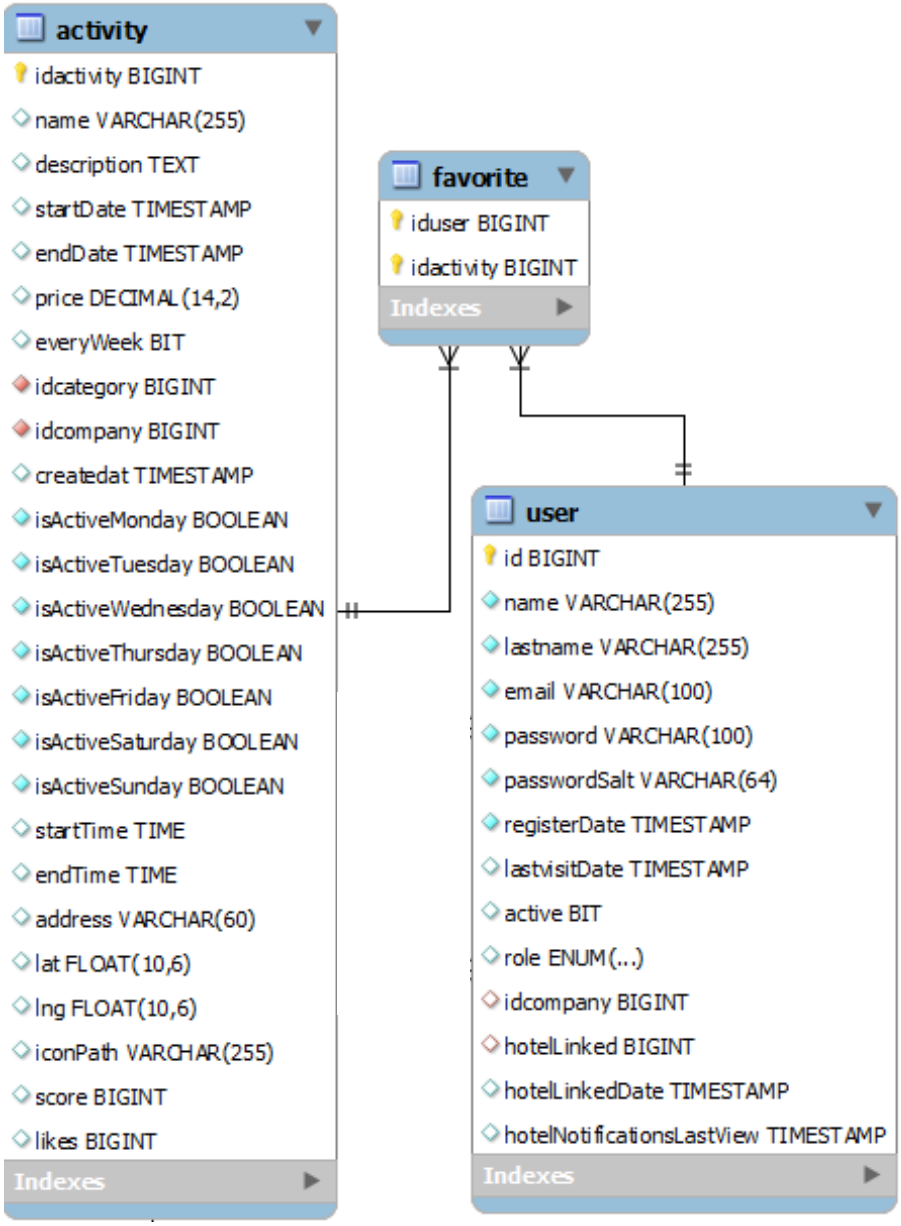


Figura. 7.27. Base de datos. Interrelación activity-favorite-user

En estas relaciones podemos encontrarnos dos relaciones de uno a muchos entre la tabla *activity* y *favorite*, y entre *user* y *favorite*. En la tabla *favorite* las claves principales son a su vez dos claves foráneas de las tablas *activity* y *user*. Estas filas permiten almacenar las referencias de las actividades favoritas de cada usuario.

Activity-favorite-user.

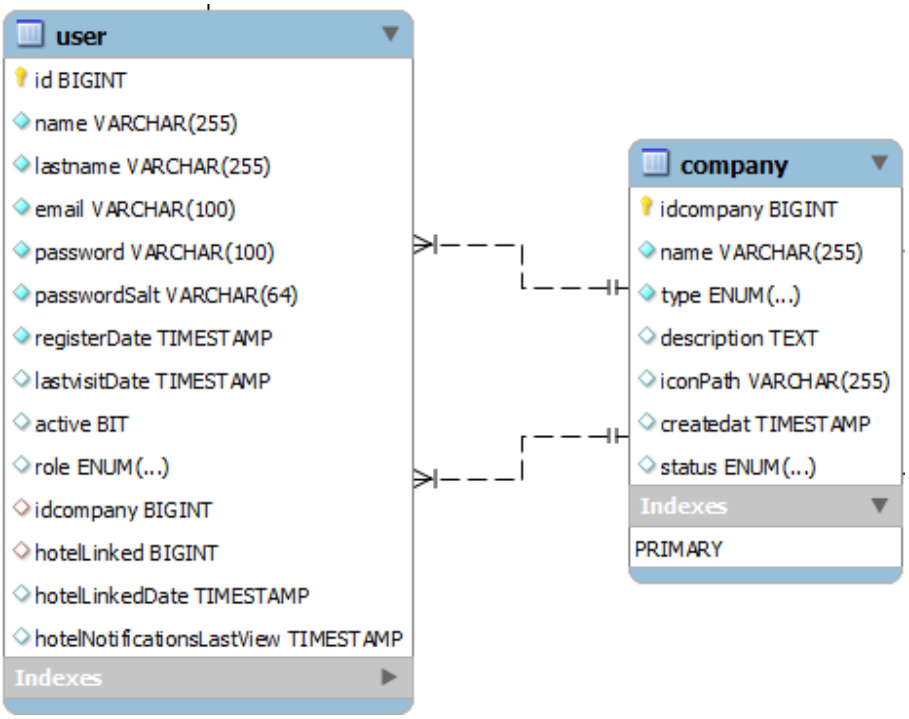


Figura. 7.28. Base de datos. Interrelación user-company

Entre estas dos tablas se aprecian dos interrelaciones de uno a muchos. Esto es debido a que el usuario puede tener relación con una empresa por dos motivos diferentes.

La primera relación sirve para asociar un usuario, con rol como cliente, a un hotel, es decir, asocia el campo *hotelLinked* de la tabla *user* al campo *idcompany* de la tabla *company* (compañía tipo hotel).

La segunda relación comprende a *idcompany* de la tabla *user* con *idcompany* de la tabla *company*. Esta relación asocia a un usuario con una empresa para que ese usuario pueda ser gestor o administrador y realizar las funciones correspondientes.

7.2.12.- Modelo de despliegue.

Cuando el sistema es finalizado ha de ser desplegado en los componentes físicos (hardware) que deben cumplir unos requisitos software.

Podemos observar el modelo de despliegue en el siguiente diagrama:

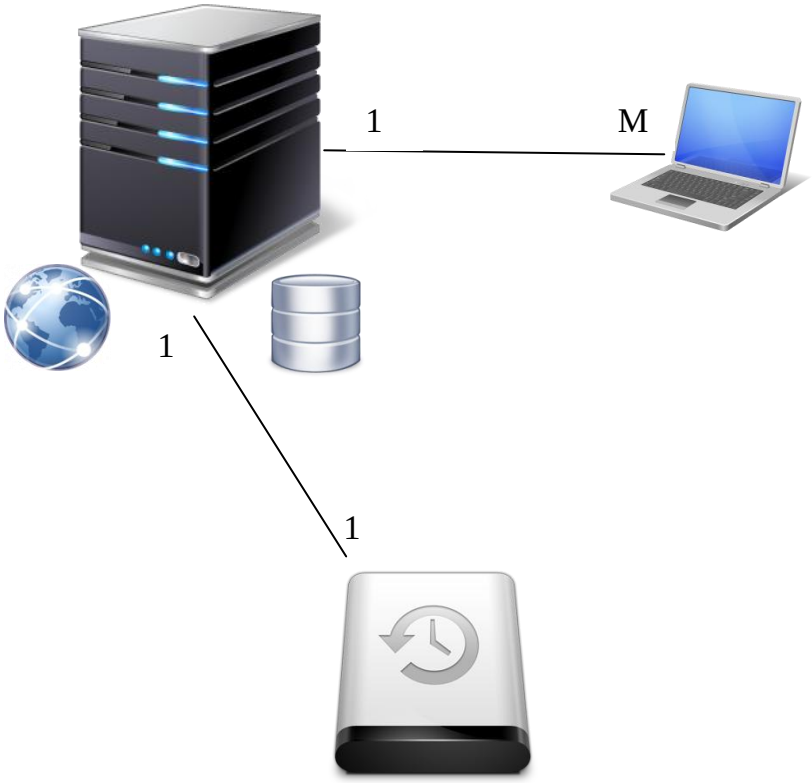


Figura. 7.29. Modelo de despliegue

Servidor principal

Requisitos hardware:

- Procesador: 2x Intel® Xeon® E7-4820, 8C, 2.00GHz, 18M Cache, 5.86GT/s, 105W TDP, Turbo, HT, DDR3-980MHz
- Memoria: 16GB Memory for 2 CPUs, 1066MHz (4x4GB 2R LV RDIMMs), 2 Memory Risers, 1333MHz DIMMs
- Almacenamiento: 4x 1TB, SATA, 2.5-in, 7.2K RPM Hard Drive
- Tarjeta controladora RAID: PERC H700 Integrated RAID Controller, 512MB Cache
- Conectividad RAID: RAID5 for PERC H200/H700, 4 HDDs
- Tarjetas de red: Intel Gigabit ET Dual Port Server Adapter, Cu, PCIe-4

Requisitos software:

- SO: Ubuntu Server
- Servidor web: Apache + PHP
- Servidor de bases de datos: MySQL
- Copias de seguridad: Bacula

Servidor de respaldo

Requisitos hardware:

- Procesador: Intel® Xeon® E3-1220, 4C/4T, 3.10GHz, 8M Cache, 80W TDP, Turbo
- Memoria: 4GB Memory (1x4GB), 1600Mhz, Dual Ranked, Low Volt UDIMM
- Almacenamiento: 2x 1TB, SATA, 2.5-in, 7.2K RPM Hard Drive
- Tarjeta controladora RAID: PERC H200 Integrated RAID Controller
- Conectividad RAID: RAID1 for PERC H200/H700, 2 HDDs
- Tarjetas de red: Intel® PRO/1000PT GbE Single Port Server Adapter, Cu, PCIe-1

Requisitos software:

- SO: Ubuntu Server
- Copias de seguridad: Bacula

Clientes

Navegadores (versiones más recientes)

- Mozilla Firefox
- Safari
- Google Chrome
- Opera

Aplicación para Android

Aplicación desarrollada en este proyecto para interactuar con el sistema de forma personalizada.

Aplicación para Karotz

Aplicación desarrollada en este proyecto para interactuar con el sistema.

7.2.13.- Diseño arquitectónico.

Arquitectura de la solución.

La arquitectura en su capa más superior es tipo cliente-servidor. Este tipo de arquitecturas están basadas en que en uno de los lados se encuentra un servidor, el cual es único, y uno o varios clientes. Los cliente envían solicitudes de algún tipo de recursos al servidor y este último procesa las solicitudes y responde a los clientes.
A continuación se muestra una figura donde se puede observar la arquitectura:

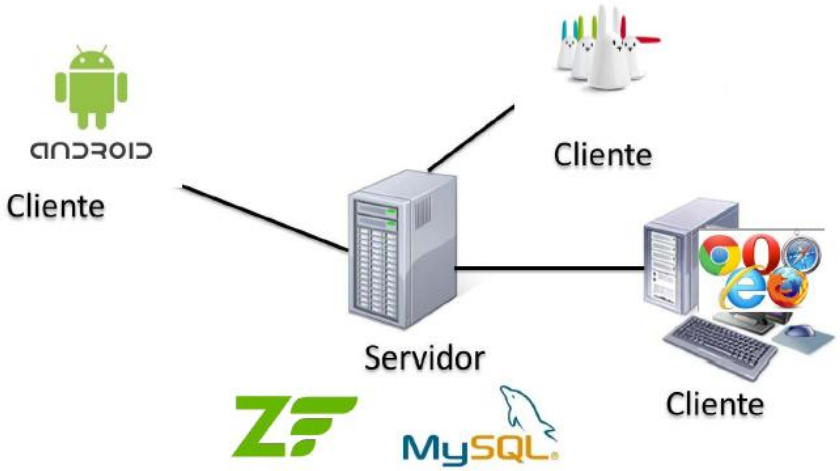


Figura. 7.30. Arquitectura de la solución

Características propias de esta arquitectura:

- El servidor puede dar respuesta a varios clientes a la vez.
- Cada cliente puede manejar los datos de un usuario concreto.
- El cliente web no depende del SO que se esté ejecutando.
- El cliente al ser una especie de interfaz de los datos, cualquier cambio en el servidor o en la base de datos no tendrán mayor repercusión sobre los clientes.

En función del tipo de cliente se comunicará con el servidor por una vía u otra, el cliente web accederá a los servicios del servidor mediante un portal web. Los clientes robóticos y Androides realizarán peticiones y recibirán las respuestas a través de una API.

Funciones de los clientes:

- Mostrar los datos al usuario, como actividades y hoteles.
- Enviar datos al servidor como nuevos usuarios, actividades y hoteles.
- Procesar los efectos de la interfaz.

Funciones del servidor:

- Procesar las peticiones que realizan los clientes.
- Validar los formularios.
- Realizar los cálculos para el sistema de recomendación
- Realizar operaciones CRUD a la base de datos.
- Devolver respuestas a los clientes.

Arquitectura de la aplicación.

Este proyecto está diseñado bajo el patrón modelo-vista-controlador impuesto por el framework utilizado y que en conjunción con la arquitectura cliente-servidor de PHP se divide en las siguientes capas:

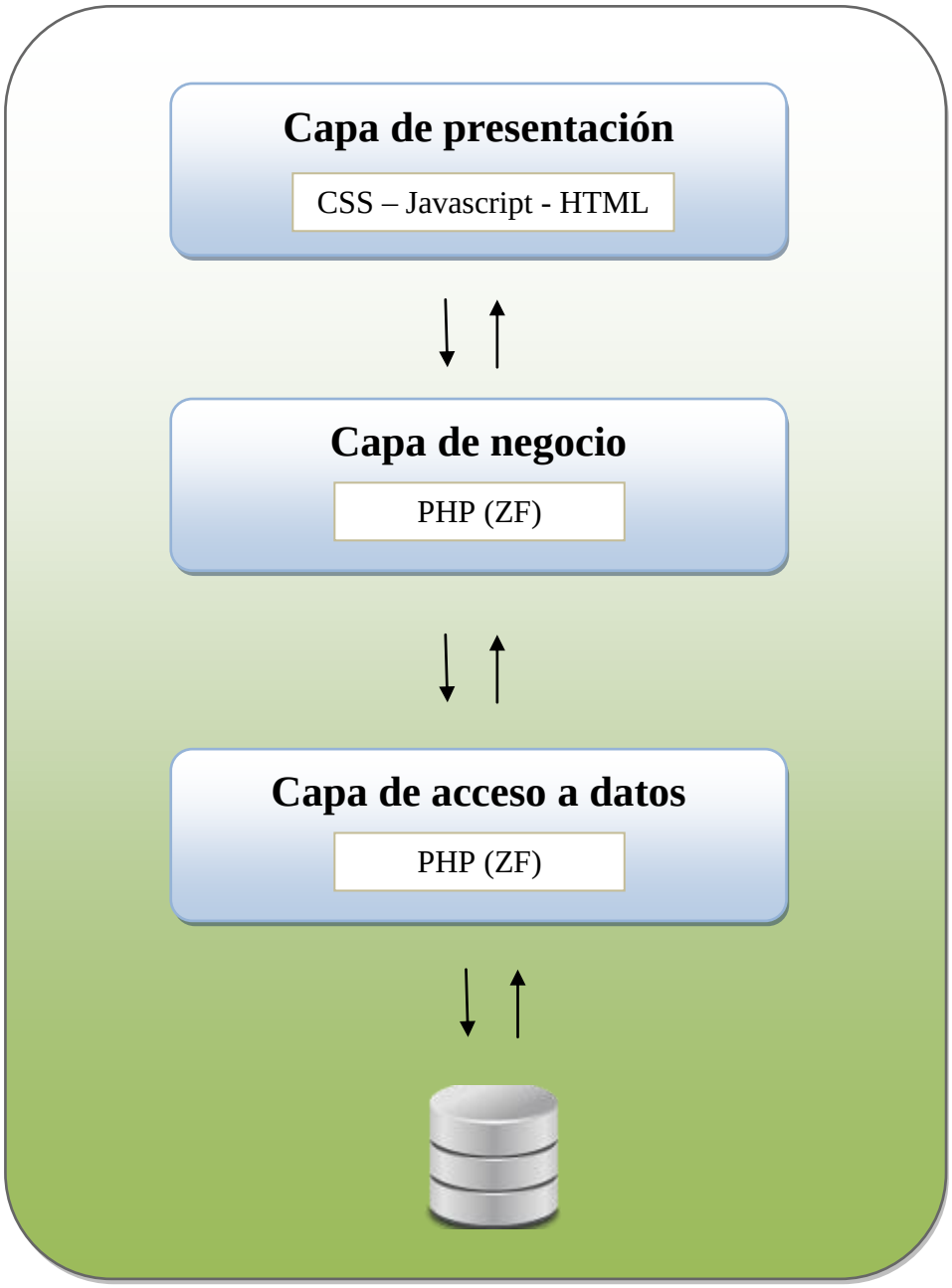


Figura. 7.31. Arquitectura servidor

Capa de presentación: Es la capa más superior de este diagrama y representa la interfaz de usuario de la aplicación. Esta interfaz contiene código HTML y JavaScript. El formato de presentación viene dado por las hojas de estilo CSS de la aplicación. La función de esta capa es la de interactuar con el usuario, permitiéndole que se comunique con el sistema en ambos sentidos.

Capa de negocio: Esta capa se encarga de gestionar las peticiones del usuario. Se ejecutan los procesos asociados a cada petición y respuesta, y hace de intermediaria entre la capa de presentación y la capa de acceso a datos. En esta parte es donde está toda la lógica de la aplicación.

Capa de datos: En la capa más inferior nos encontramos con una zona que es donde se almacenan todos los datos del sistema. Contiene métodos y funciones externas al sistema pero que interactúan con este, concretamente con la capa inmediatamente superior, la capa de negocio. El gestor de base de datos se encarga de encolar las peticiones de datos, procesarlas y dar respuesta al sistema.

Zend-Framework y su arquitectura.

Introducción a Zend-Framework.

Para el desarrollo del sistema se ha utilizado Zend-Framework. Es un framework de desarrollo de aplicaciones y servicios web con PHP 5. ZF está implementado totalmente orientado a objetos y la estructura de sus componentes es idónea porque cada componente posee una baja dependencia del resto. De esta manera en función de la aplicación a desarrollar sólo es necesaria una parte de la librería de Zend.

Aunque se pueden utilizar de forma individual, los componentes de la biblioteca estándar de Zend Framework conforman un potente y extensible framework de aplicaciones web al combinarse. ZF ofrece un gran rendimiento y una robusta implementación MVC, una abstracción de base de datos fácil de usar, y un componente de formularios que implementa la prestación de formularios HTML, validación y filtrado para que los desarrolladores puedan consolidar todas las operaciones usando de una manera sencilla la interfaz orientada a objetos. Otros componentes, como Zend_Auth y Zend_Acl, proveen autentificación de usuarios y autorización diferentes a las tiendas de certificados comunes. También existen componentes que implementan bibliotecas de cliente para acceder de forma sencilla a los web services más populares.

A continuación se muestra una figura donde se observan los componentes del framework. Estos componentes se pueden agrupar en seguridad, MVC, Internacionalización, Datos, Servicios web y el núcleo:

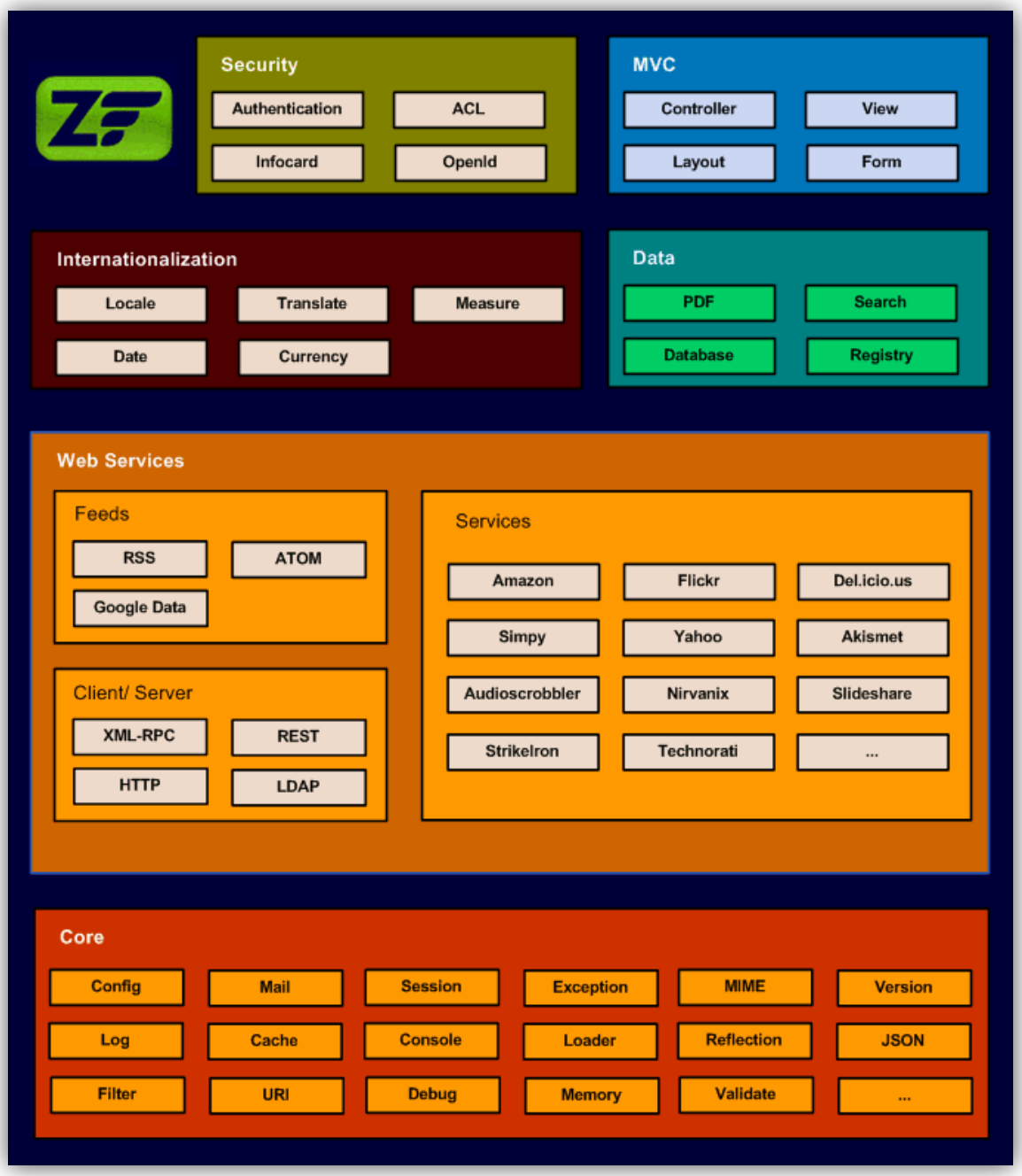


Figura. 7.32. Módulos de Zend Framework

Arquitectura de Zend Framework. Modelo-vista-controlador.

La mayoría de las aplicaciones web se estructuran de la siguiente forma: presentación o vista, lógica de negocio y acceso a datos. El patrón MVC separa estos tres conceptos y establece las relaciones entre ellos. Fue introducido por Trygve Reenskaug en los años 70 aunque no fue hasta hace no mucho cuando se comenzó a conocer de manera popular.

- **Modelo:** Esta es la parte de la aplicación que define la funcionalidad básica de un conjunto de abstracciones de datos. En él se encuentran las rutinas de acceso a datos y alguna lógica del sistema. En función del sistema el modelo puede realizar un mapeo total de los objetos de la base de datos a objetos de php.
- **Vista:** Las vistas definen exactamente lo que se va a presentar al usuario. Normalmente los controladores pasan datos a cada vista para después estas mostrarlas con algún formato. Aquí es donde se encuentra HTML mezclado con PHP en las aplicaciones MVC.
- **Controlador:** El controlador es la parte que une todo el patrón junto. Se encargan de la manipulación de los modelos, de la decisión de la vista que se muestra al usuario en función la petición recibida y pasan a la vista los datos que esta necesita, entre otras tareas. En el interior de los controladores están definidas las acciones para ese controlador. Habitualmente los controladores reflejan objetos de la vida real como pueden ser los usuarios. De esta manera un controlador de usuarios tendrá típicamente acciones como loguearse, salir del sistema, modificar sus datos, etc.

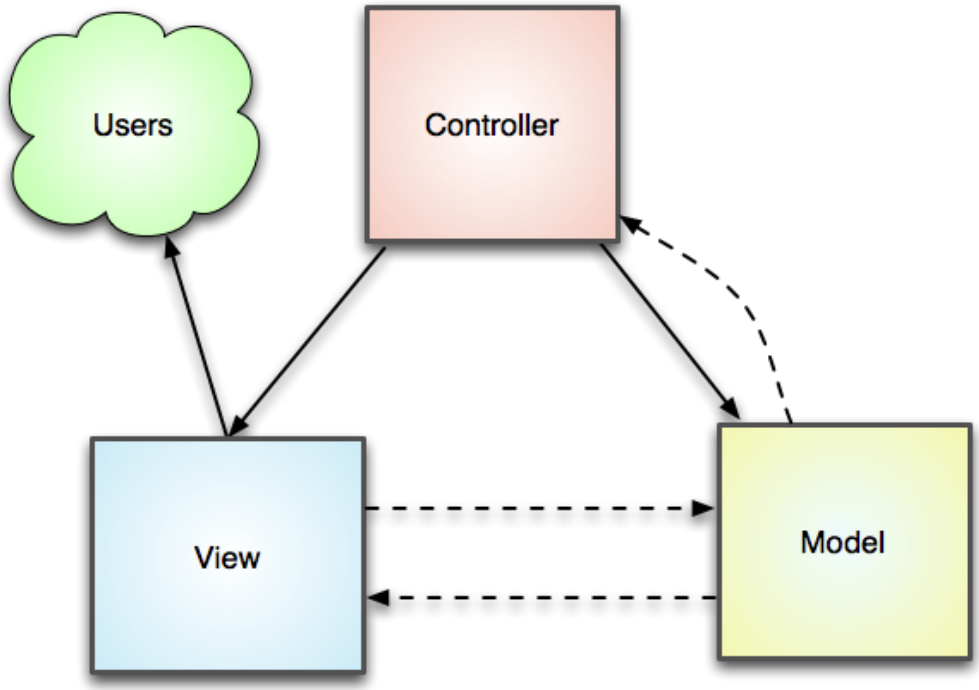


Figura. 7.33. Arquitectura modelo-vista-controlador

Interacción de los componentes de una aplicación bajo el patrón MVC:

1. El usuario interactúa con la interfaz de usuario.
2. El controlador la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos (handler) o callback. En el caso de Zend Framework ese gestor de eventos se llama Bootstrap.
3. El controlador accede al modelo, modificándolo de forma adecuada a la acción solicitada por el usuario. Los controladores complejos están a menudo estructurados usando un patrón de comando que encapsula las acciones y simplifica su extensión.
4. El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se reflejan los cambios en el modelo. El modelo no debe tener conocimiento directo sobre la vista.
5. La interfaz de usuario espera nuevas interacciones del usuario y de nuevo comienza el ciclo.

Arquitectura de Zend Framework. Elementos de trabajo.

- Bootstrap: La clase Bootstrap es usada para cargar los componentes o recursos comunes que son usado por todos o la mayoría de los controladores, vistas y modelos de la aplicación. Las tareas comunes que ejecuta son conexión con la base de datos, inicio de sesión carga de archivos de configuración y carga de librerías.
- Plugins: La arquitectura del controlador incluye un sistema de plugins que pueden ser llamados cuando ocurren eventos durante el proceso de vida de la acción. Estas llamadas se producen antes o después del proceso de ruteo, antes o después del bucle de inicio de la acción, y antes o después de la ejecución de la acción. La aplicación de los plugins habitual está enfocada a librerías como control de acceso de usuarios, navegación en el sistema y configuración del layout.
- Helpers: Los helpers o action helpers permite inyectar en el tiempo de ejecución de la acción una funcionalidad bajo demanda dentro de la acción, y de esta forma extender su funcionalidad. Esta funcionalidad permite que varios controladores utilicen funcionalidades comunes entre ellos sin necesidad de extenderlos. Zend Framework ya incluye varios de ellos que son muy útiles en casi todas las aplicaciones web:
    - Action Stacks: que permite enviar solicitudes al ActionStack del Front Controller. De esta forma durante la propia ejecución de la acción se puede añadir a la cola de lanzamiento nuevos plugins.
    - Autocomplete: ayuda a ofrecer resultados potenciales de las búsquedas realizadas para que se muestre en el layout como sugerencias.

- ContextSwitch and AjaxContext: facilita el cambio de contexto para ofrecer una respuesta u otra en función del formato de la petición.
- FlashMessenger: este helper permite pasar mensajes para la siguiente petición.
- JSON: convierte rápidamente a este formato la respuesta elegida. De esta forma se puede combinar con las llamadas AJAX que esperan este tipo de conjunto de datos.
- Redirector: es un elemento que cubre la necesidad de realizar redirecciones a dentro de la aplicación o cualquier otra URL.
- ViewRenderer: está diseñado para eliminar la necesidad de instanciar vistas con controladores, crear una vista que sea despachada por todos los controladores, mostrar una vista determinada sin ninguna intervención y especificar cualquier parámetro de una vista que vaya a ser mostrada.

Arquitectura de Zend Framework. Flujo de ejecución.

En la arquitectura de Zend Framework nos encontramos con un elemento que se encarga de gestionar todo el proceso de dispatch, el Front Controller.

1. Se inicia el Bootstrap de la aplicación, que registra cada componente del sistema.
2. El Front Controller toma el mando. En ese instante se entra en un bucle donde se rutea la petición y uno por uno los plugins son llamados.
3. Tras la llamada al plugin se inicia la acción requerida.
4. La acción se ejecuta y lanza cada helper asociado o registrado en la acción.
5. Si existen más plugin por lanzarse se vuelve al punto 3.
6. Se envía la respuesta.

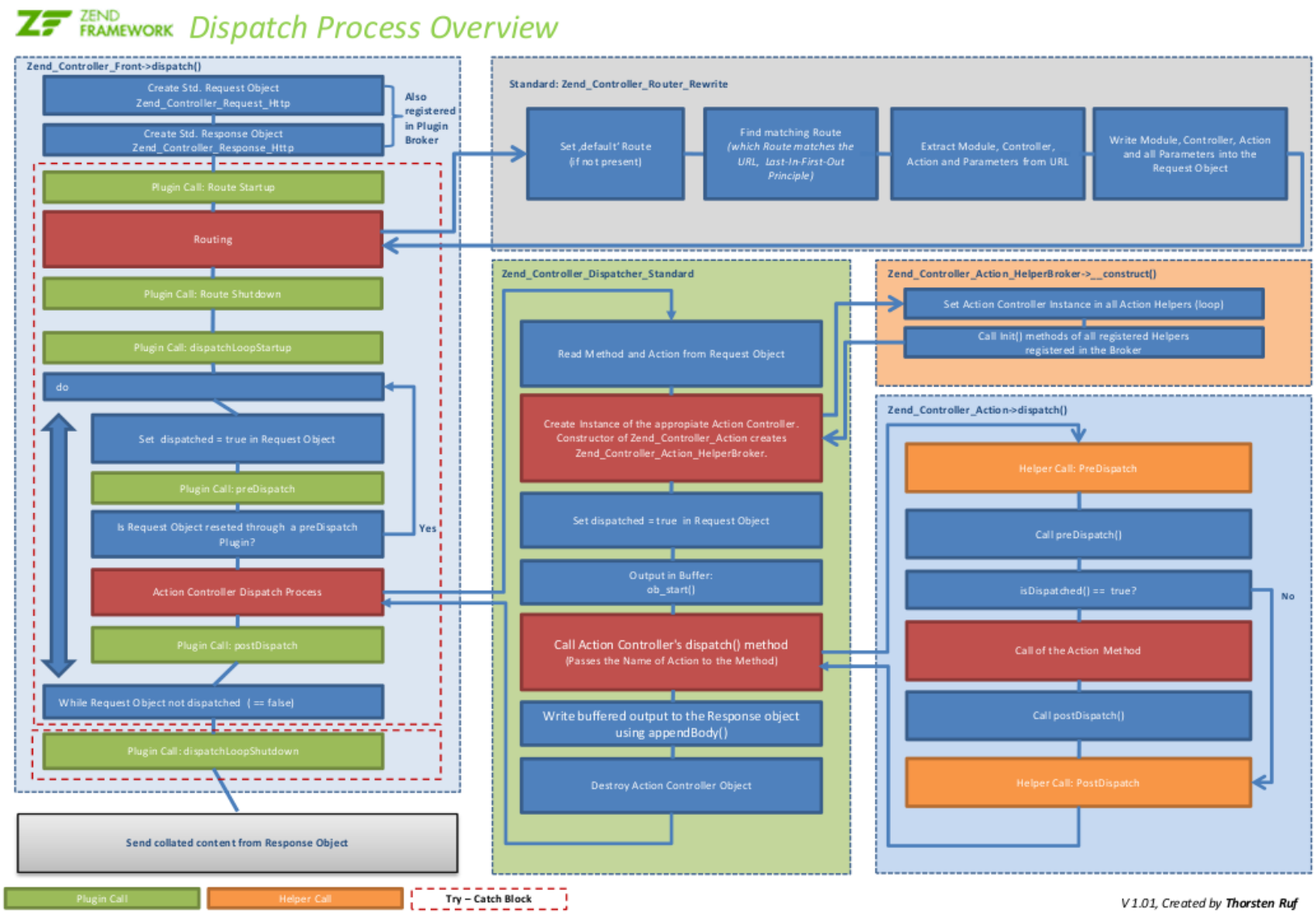


Figura. 7.34. Flujo de ejecución en Zend Framework

7.2.14.- Detalles de la implementación.

Componentes de Zend-Framework utilizados para el desarrollo del sistema:

Nombre del componente	Uso en el sistema
Zend_Acl	Listas de control de acceso. Este componente ha sido ampliamente utilizado para el desarrollo de una librería que permite la gestión de usuarios. Gestiona todo lo relacionado con el acceso y rol del usuario.
Zend_Auth	Maneja la autorización de los usuarios. Junto con Zend_Acl ha permitido el desarrollo de la librería de gestión de usuarios. Mantiene la sesión del usuario logueado.
Zend_Captcha	Este componente genera los captcha para los formularios.
Zend_Controller	Contiene todos los elementos de los controladores de la estructura.
Zend_Date	Utilizado para realizar operaciones con las fechas durante el procesado de formularios.
Zend_Db	Importante componente utilizado en cada controlador. Sus subclases permiten el acceso a la base de datos y realizar cualquier tipo de consultas filtrando los datos de las querys y pudiendo realizar un cacheado de los datos obtenidos.
Zend_Debug	Utilizado para realizar debugging en el sistema.
Zend_File	Para la carga y procesado de archivos en el sistema mediante formularios.
Zend_Filter	Permite el filtrado de valores en los formularios del sistema.
Zend_Form	Este componente ha sido también en repetidas ocasiones del sistema para confección cada uno de los formularios que aparecen. Permite una sencilla reutilización, validación y acceso a parámetros.
Zend_Http	Utilizado para el acceso a las peticiones o respuestas del servidor y sus parámetros.

Zend_Json	Componente que permite convertir los datos a formato json. Utilizado en las respuestas de la API .
Zend_Layout	El uso de este componente ha permitido modificar los parámetros de la interfaz principal.
Zend_Navigation	Usado para crear los elementos de la navegación. Este componente se ha combinado con Zend_Acl para el usuario experimente los diferentes menús de navegación en función del rol al que pertenezca.
Zend_Paginator	Ha permitido realizar una paginación de los elementos a mostrar en el sistema. Tiene varios tipos de navegación en la paginación y es posible cachear los elementos de paginados.
Zend_Registry	Se llama así a este componente que ha permitido mantener variables en el sistema durante la ejecución de una llamada en una página.
Zend_Rest	Permite desarrollar una API Rest de manera sencilla para que puedan acceder a los servicios del sistema.
Zend_Test	Este componente ha permitido realizar pruebas unitarias en las diferentes implementaciones del sistema. Se testea bajo PHPUnit.
Zend_Uri	Componente que da soporte para la manipulación de URLs en el sistema.
Zend_Validate	Ha sido clave para reducir el tiempo de desarrollo, ya que el uso de este componente permite validar prácticamente cualquier tipo de formulario gracias a todas las funciones de validación que posee.
Zend_View	Un componente básico para el desarrollo del sistema. Permite la gestión de todas las vistas de los controladores.

Tabla 7.4. Componentes de Zend-Framework utilizados

Estructura de la aplicación.

La estructura de la aplicación está realizada bajo las recomendaciones de estructuración de Zend Framework, dividiendo la aplicación en módulos y a su vez en modelos, vistas y controladores. Además de estos, se añaden las librerías propias, plugins y helpers.

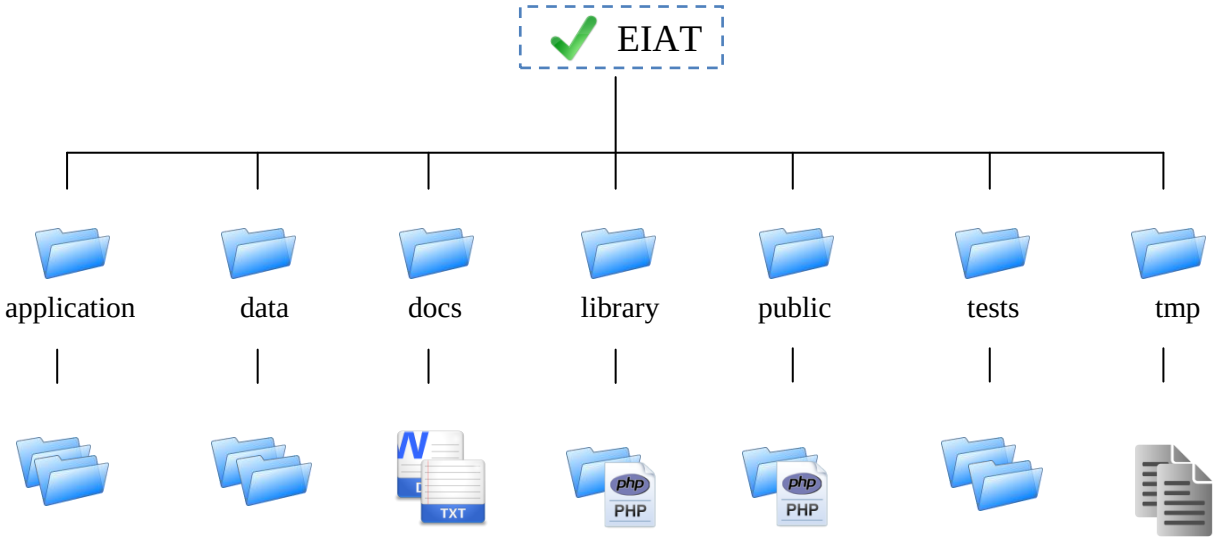


Figura. 7.35. Estructura de la aplicación. Nivel cero

En la planta inferior de la aplicación nos encontramos con estas siete carpetas que contienen toda la aplicación excepto los datos persistentes.

- **Application:** Esta carpeta contiene toda la lógica de la aplicación a excepción del núcleo de ZF y las librerías añadidas.
- **Data:** contiene los datos persistentes de la aplicación que no corresponden a la base de datos, como pueden ser: la memoria cache del paginado de búsquedas y las sesiones de usuarios.
- **Docs:** En este directorio debe estar el manual de instalación de aplicación, así como su la documentación de su API, notas de proyecto y documentación de php.
- **Library:** Contiene el núcleo de Zend Framwork y otras librerías propias relativas al proyecto. En este caso se ha desarrollado una librería para la gestión del acceso a usuarios y el sistema de recomendación.
- **Public:** Esta será la carpeta pública para el servidor de PHP. En ella se encuentra el fichero index.php que ejecuta toda la aplicación. Además podemos encontrar otras carpetas como CSS y JavaScript.
- **Tests:** Esta carpeta está destinada a contener los ficheros con los que se realizan los tests al sistema. Para ello Zend Framework provee de un "bootstrap" para probar cada acción con pruebas unitarias utilizando la librería PHPUnit.
- **Tmp:** Carpeta donde se encuentran los ficheros temporales de la aplicación.

Bajo la carpeta *application* se define una estructura donde a primera vista ya se observa el patrón modelo-vista-controlador:

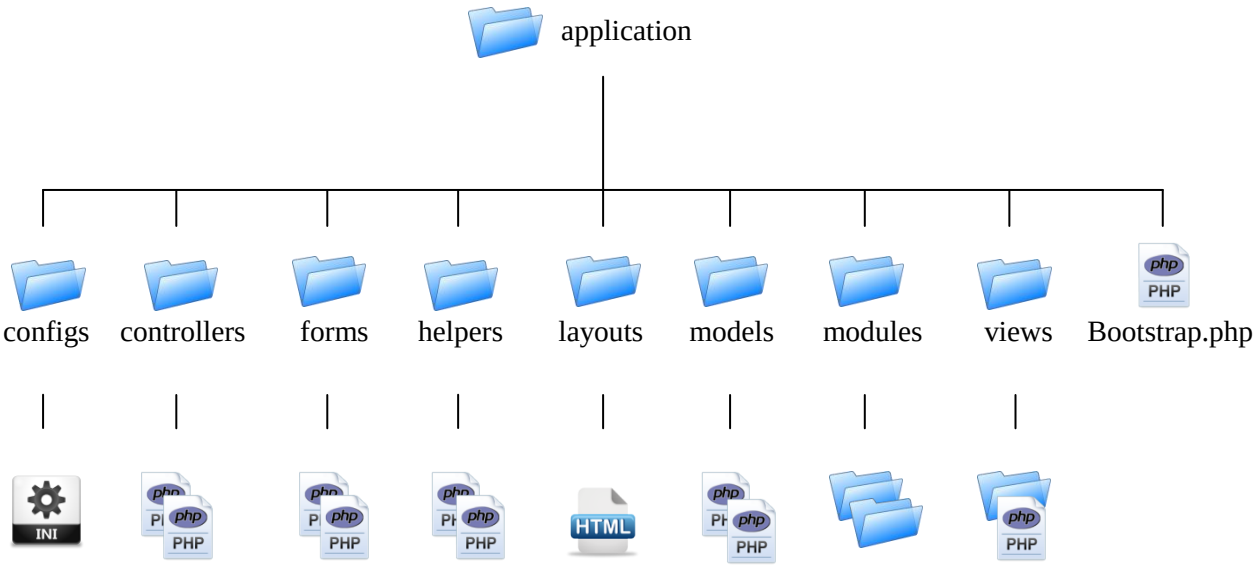


Figura. 7.36. Estructura de la aplicación. Application

- **Configs:** Contiene el fichero application.ini donde se encuentra toda la configuración inicial de la aplicación. Aquí se define la rutas globales de la aplicación, las credenciales del gestor de base de datos y otros parámetros.
- **Controllers:** Alberga todos los controladores de este módulo del sistema, esto es, actividades, categorías, compañías, hoteles, usuarios y controlador principal de acceso.
- **Forms:** Este directorio almacena todos los formularios que utiliza este módulo del sistema. De esta manera desde cualquier controlador es posible instanciarlo y reutilizarlo. Aquí se encuentran los formularios de login, registro, búsqueda de actividades, búsqueda de hoteles, cambios de contraseña y otros.
- **Helpers:** Para este proyecto se ha creado un helper, *display_helper*, que maneja qué módulos se muestran en la interfaz basados en el rol del usuario.
- **Layouts:** En el interior se encuentra el fichero HTML que estructura la interfaz de usuario. Pueden haber varios y ser seleccionados en tiempo de ejecución.
- **Models:** Los modelos del sistema es la capa de acceso a datos. Mantiene los ficheros que hacen el mapeo por defecto a determinadas tablas de la base de datos y otros ficheros que permiten hacer un mapeo personalizado de estas tablas para la aplicación.
- **Modules:** Contiene otros módulos de la aplicación.

- **Views:** En *views* están definidas cada una de las vistas del sistema, es decir, cómo una acción determinada del controlador muestra los datos al usuario. Para cada acción de cada controlador existe una vista asociada. De modo que esta carpeta se subdivide en tantas carpetas como controladores hayan, y finalmente, en el interior de cada una se encuentran las vistas de cada acción.
- **Bootstrap:** Este fichero php prepara y lanza toda la configuración en cada ejecución de la aplicación.

Continuando con el análisis de la estructura, en la carpeta *public* se encuentra el fichero de ejecución de la aplicación, *index.php* y todas las carpetas públicas que dan formato y usabilidad a la aplicación:

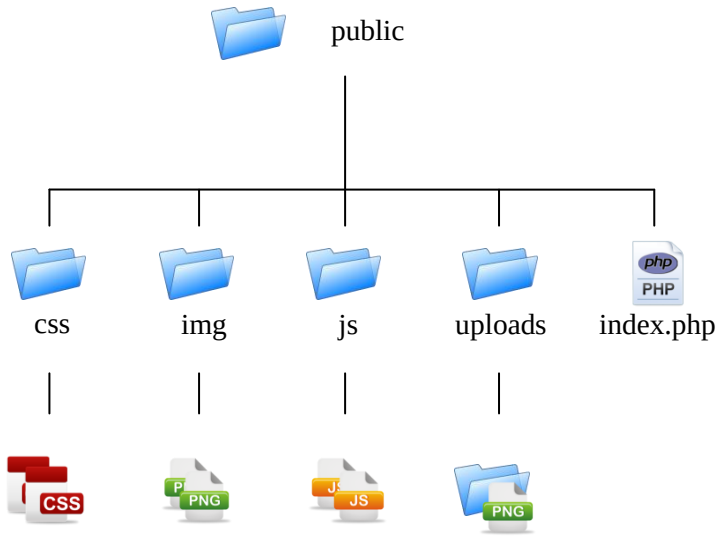


Figura. 7.37. Estructura de la aplicación. Public

- **CSS:** Esta carpeta contiene los ficheros de estilo de la aplicación que dan formato a la interfaz de usuario. Incluye los ficheros propios de la aplicación y los de la librería Bootstrap de Twitter.
- **Img:** Almacena dos paquetes de imágenes que usa la librería Bootstrap y otros ficheros de imágenes.
- **JS:** Mantiene los ficheros que dan funcionalidad extra a la aplicación. Estos ficheros pertenecen a la librería JQuery y permiten que exista una usabilidad extra en la aplicación.
- **Uploads:** En esta carpeta de la aplicación se almacenarán todos los ficheros derivados del uso del sistema, como son:
    - los iconos de hoteles y empresas
    - los iconos de las actividades
    - otros iconos de sliders, etc.

- **Index.php:** Fichero de inicio de la aplicación por donde todos los usuarios accederán a todas las funcionalidades de la misma.

En la carpeta tests es donde se encuentran los ficheros para realizar los tests de la aplicación utilizando la librería PHPUnit.

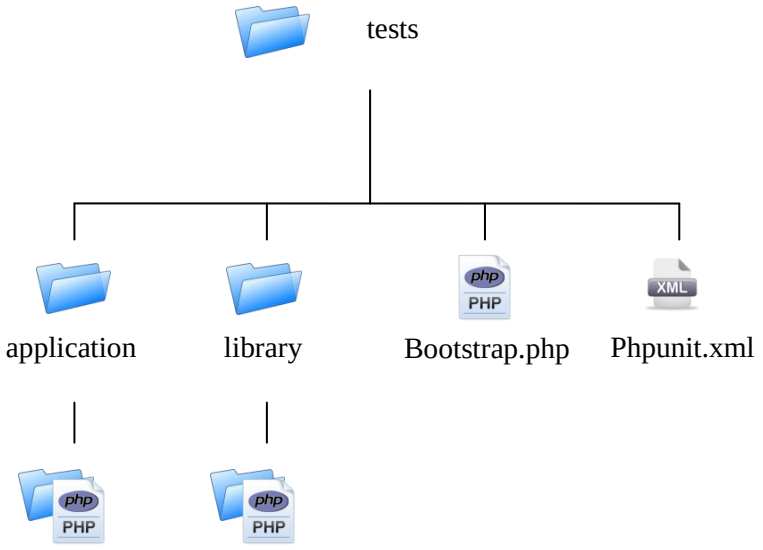


Figura. 7.38. Estructura de la aplicación. Tests

- **Application:** Contiene todos los elementos de la aplicación a testear, como los controladores. En ellos se indican que llamadas se deben ejecutar y qué respuestas debe dar el sistema para quedar validado.
- **Library:** Incluye las librerías del sistema a testear.
- **Bootstrap:** Es el fichero de inicialización del sistema cuando estamos en modo testeo
- **Phpunit.xml:** Configura PHPUnit para poder ejecutarlo en el sistema y realizar los tests.

Desarrollo de un sistema seguro de usuarios.

Autenticación de usuarios con Zend Framework.

Introducción

Para este proyecto se ha desarrollado una parte fundamental del sistema y en la que se ha invertido una gran cantidad de tiempo, la lógica de registro, acceso y gestión de usuarios. Esta librería desarrollada es exportable a otros proyectos y

permite que desde el mismo portal de entrada se de acceso a los usuarios con diferentes roles.

Para desarrollar un sistema de usuarios seguro se ha seguido las recomendaciones de Zend Framework y se ha utilizado su API manejar las sesiones de usuario.

Todo sistema basado en usuarios debe almacenar en algún lugar (B.B.D.D.) los datos cada perfil. En este proyecto se ha creado una tabla con este propósito. Los campos que harán de identidad y credencial de cada usuario serán el correo electrónico y su clave personal. No pueden existir dos correos iguales en el sistema y la clave debe tener una longitud entre 6 y 24 caracteres.

Seguridad en el sistema con salting

En criptografía, la sal comprende bits aleatorios que son usados como una de las entradas en una función derivadora de claves. La otra entrada es habitualmente una contraseña. La salida de la función derivadora de claves se almacena como la versión cifrada de la contraseña. La sal puede también ser usada como parte de una clave en un cifrado u otro algoritmo criptográfico. La función de derivación de claves generalmente usa una función hash. A veces el vector de inicialización, un valor generado previamente, es usado como sal.

Supongamos que la clave secreta de un usuario es robada, y se sabe que usa una de 200.000 palabras inglesas como contraseña. El sistema usa una sal de 32 bits. La clave salada es ahora la contraseña original unida a una sal aleatoria de 32 bits. A causa de esto, los hashes precalculados del atacante carecen de valor. Deberá calcular el hash de cada palabra junto con cada una de las 232 (4,294,967,296) posibles combinaciones de sales hasta que se encuentre una coincidencia. El número total de posibles entradas se puede obtener multiplicando el número de palabras en el diccionario por el número de posibles sales:

$$2^{32} x 200000 = 8.58993459 x 10^{14}$$

Para completar un ataque por fuerza bruta, el atacante deberá ahora computar cerca de 860 billones de hashes, en lugar de sólo 200.000. Incluso cuando la contraseña en sí misma sea simple, la sal secreta hace que romper la contraseña sea radicalmente más difícil.

Otro de los beneficios de este tipo de implementaciones es que es bien conocido que los usuarios en internet re usan sus claves para distintas cuentas y esto puede suponer un problema. Gracias a que el salting varía de una aplicación web a otra, las bases de datos no almacenan la contraseña original del usuario sino que guardan una transformación de ella y de esta manera su contraseña nunca quedará al descubierto.

La implementación concreta en este proyecto contempla la concatenación de un vector de 32 bytes (caracteres generados aleatoriamente) con la contraseña introducida por el usuario. Finalmente el resultado de esa concatenación es resumido por la función hash md5 y almacenado en la base de datos. Asimismo, para poder posteriormente realizar la autenticación de usuarios también se guarda el vector de sal creado para ese usuario específico.

En el siguiente gráfico podemos ver como se almacenan las credenciales en la base de datos del sistema:

email	password	passwordSalt
prueba@domain.com	ca905edf58d588995482dc33a09d5a43	bXXr+o?:bJeolusk:0DrKc?82
elena@domain.com	b5dda276bbae96081beeda3149b11500	,v9aTayD+W31SxM1$Xgz@tOZ]
otro@domain.com	b29e7599dcb64a64b7d80aa1fe8d7555	coP/4t2hf,x_:/M59'$9aK?-#GVa6QN9
diego@domain.com	fc6ba55a8728b6fbcead9b7886316f75	%N*k"$u!0{BEoeV0r>-$Z0.7v:2O*)C!
x@x.com	178d850da9405463fe0d7475f72e9022	-7Zx5BP~o=/Kl'0"N+ZvdG)ppHc%d+$6
juan@senderismocabrera.com	3eb793e98c52962d2e22ba2289021cdc	.5*DP|"GNydE;z0v[Co%e"aH>83+Yw?c
g@x.com	6d7c6fd433d102691b41b9b69a0e6666	DrB{m,}2bc/`[?5r_5#gJ(wk$/fz%.nb

Figura. 7.39. Campos y contenido de los usuarios en la base de datos

Los usuarios pueden realizar estas tres acciones:

- Registrarse
- Loguearse
- Salir

A continuación se detallará cada uno de los procesos desarrollados en el sistema.

Registro de un usuario.
Durante el proceso de registro los usuarios deben proporcionar al sistema una dirección de correo electrónico, que debe ser única en la base de datos, y una contraseña. El correo electrónico de cada usuario cumplirá la función de identificador único.

Mediante el siguiente diagrama de flujo se explicarán los pasos para que un nuevo usuario quede registrado en la base de datos:

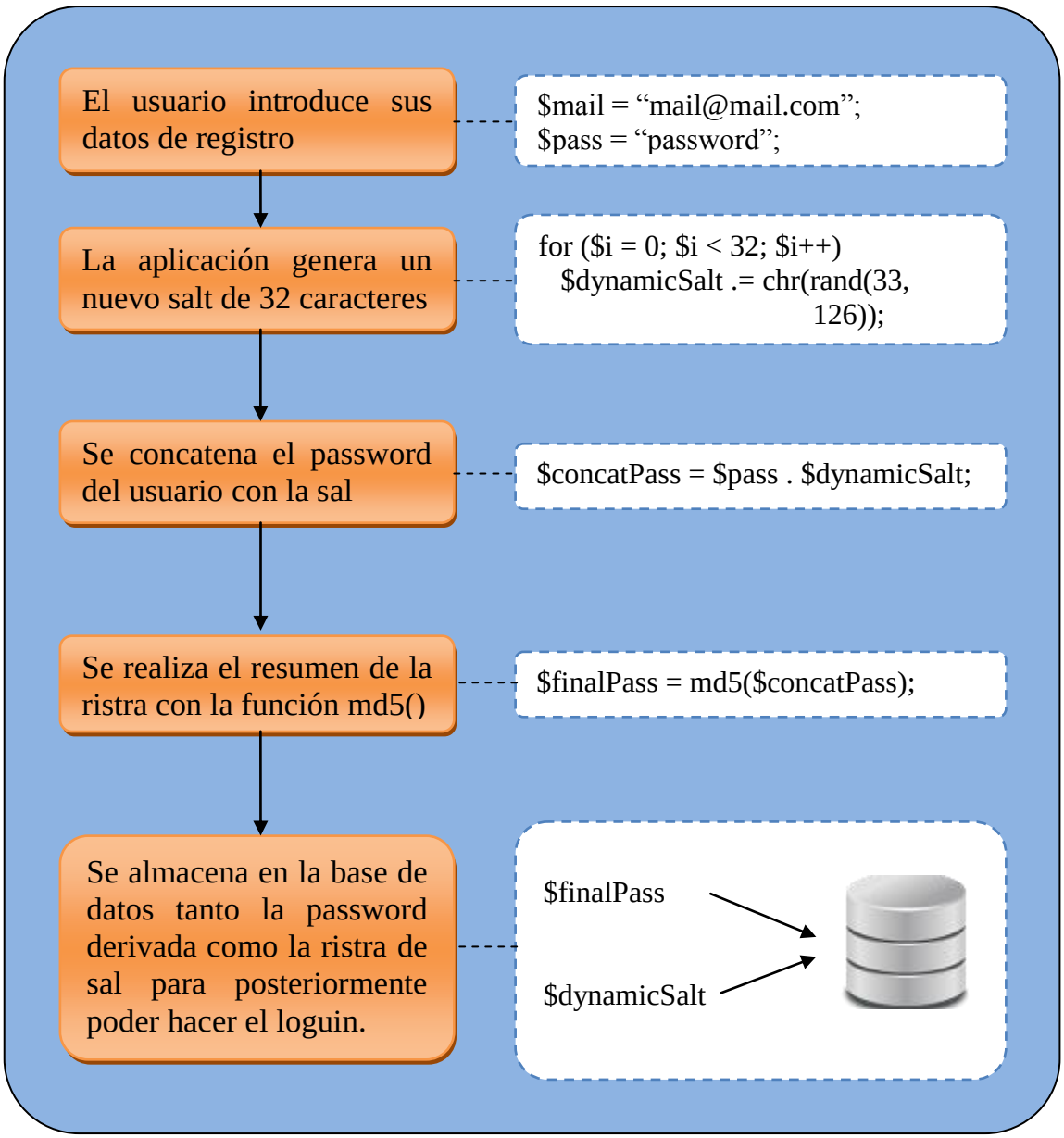


Figura. 7.40. Proceso de registro

El código abreviado correspondiente utilizando la API Zend_Auth de Zend Framework es el siguiente:

```
for ($i = 0; $i < 32; $i++)
   $dynamicSalt .= chr(rand(33, 126));
$data = array(
    'name'              => $registerForm->getValue('name'),
    'lastname'         => $registerForm->getValue('lastname'),
    'email'               => $registerForm->getValue('email'),
    'password'                            =>   md5($registerForm-
>getValue('password') . $dynamicSalt),
    'passwordSalt'  => $dynamicSalt,
);

$users = new Application_Model_DbTable_User();
$users->insert($data);
```

Figura. 7.41. Código de registro en el sistema

Loguearse

Para el proceso de login se realizan una serie de pasos similares a los del proceso de registro. Esta acción del sistema debe comparar que los datos que el usuario está proporcionando son equivalentes a uno de los datos que pertenecen a un usuario registrado. El sistema busca una coincidencia con la identidad del usuario, la dirección de correo electrónico, y si es así realiza los cálculos necesarios para comprobar si las credenciales son equivalentes.

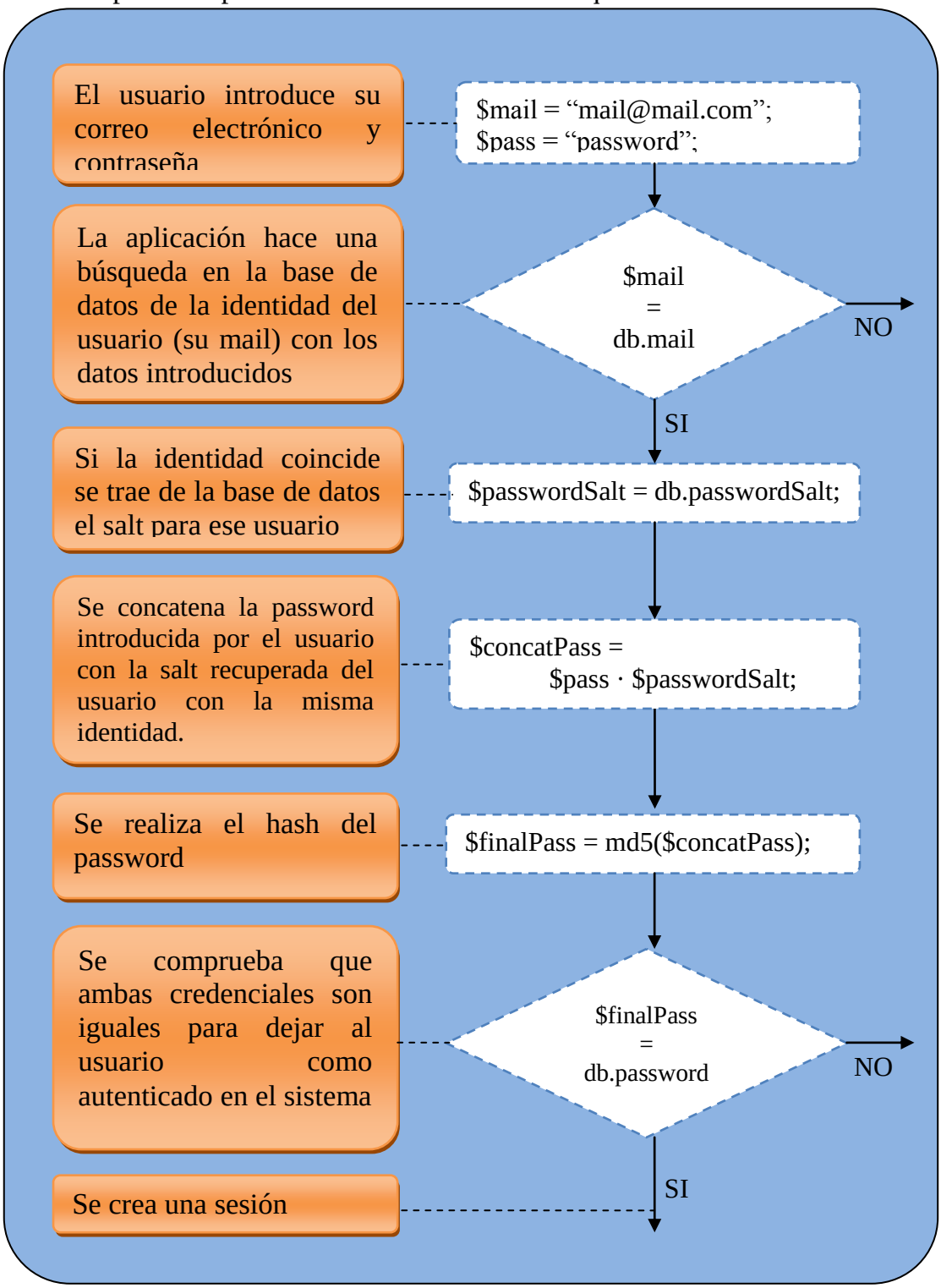


Figura. 7.42. Proceso de login en el sistema

El código correspondiente al login utilizando Zend_Auth es el siguiente:

```
$adapter = new Zend_Auth_Adapter_DbTable(
                      null,
                      'user',
                      'email',
                      'password',
                      'MD5(CONCAT(?, passwordSalt))'
            );

$adapter->setIdentity($loginForm->getValue('email'));
$adapter->setCredential($loginForm->getValue('password'));

$auth = Zend_Auth::getInstance();
$result = $auth->authenticate($adapter);

if ($result->isValid()) {
    // Login success
     $this->_redirect('/');
}
```

Figura. 7.43. Código de login en el sistema

Salir

Cuando se realiza un logout de la aplicación el sistema lo que hace es borrar la sesión que se había creado.

El código correspondiente al login utilizando Zend_Auth es el siguiente:

```
$auth = Zend_Auth::getInstance();
$auth->clearIdentity();
$auth->getStorage()->clear();
$this->_redirect('/');
```

Figura. 7.44. Código de logout

Desarrollo de un sistema autorización de usuarios utilizando ACL de Zend Framework.

Las listas de control de acceso o en inglés ACL permiten a un sistema crear una separación de privilegios. Estas listas permiten determinar que permisos de acceso tiene un objeto a un recurso. De esta manera no todos los objetos tienen los mismos privilegios ni pueden acceder a los mismos recursos. Este concepto es ampliamente utilizado en aplicaciones web, como es el caso para este proyecto.

Zend Framwork tiene una librería denominada Zend_Acl que provee al desarrollador de un conjunto de elementos muy útiles para manejar listas de control de acceso.

Se definen los dos siguiente elementos:

- **Recursos:** Los recursos son aquellos objetos u elementos a los que los usuarios desean acceder, ej. "ver usuarios".
- **Roles:** Que son los perfiles que pueden tener cada uno de los usuarios, ej. "administrador".

En la aplicación web del proyecto existen una gran cantidad de recursos, tantos como la suma de acciones de todos los controladores, pero no es necesario definir todos. Es posible crear reglas de herencia a partir de los controladores.

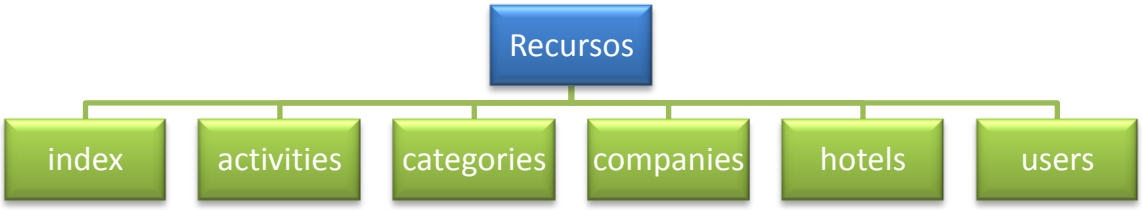


Figura. 7.45. Esquema de recursos del sistema

Definiendo los recursos en Zend Framework:

```
$this->_acl
           ->addResource('index')
           ->addResource('activities')
           ->addResource('categories')
           ->addResource('companies')
           ->addResource('hotels')
           ->addResource('users')
           ->addResource('error');
```

Figura. 7.46. Código de algunos recursos de la aplicación

A continuación hay que definir en el sistema son los roles, de nuevo podemos aplicar herencia:

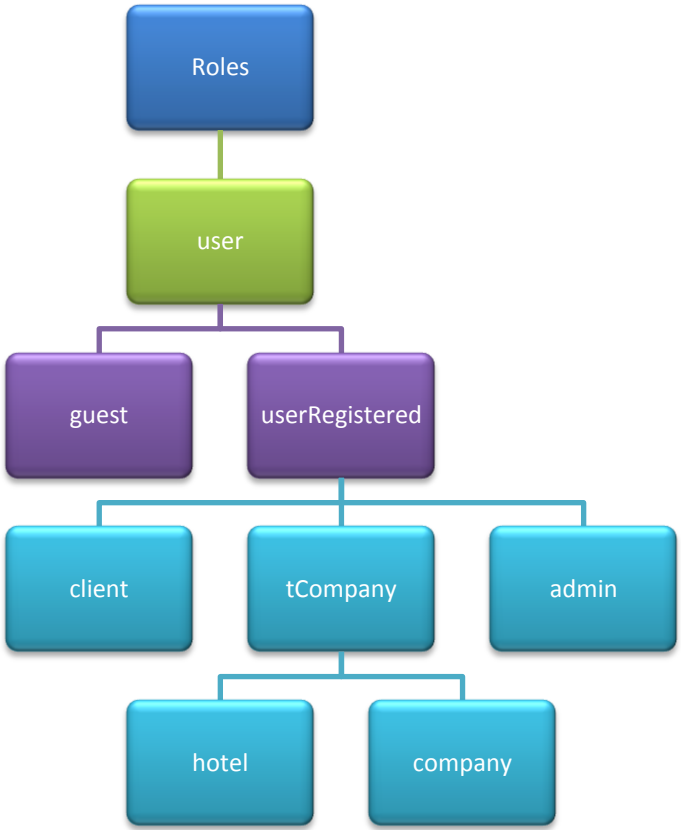


Figura. 7.47. Esquema de roles en Zend Framework

El código en PHP abreviado correspondiente es:

```
$this->_acl
           ->addRole('user')
           ->addRole('guest', 'user')
           ->addRole('userRegistered', 'user')
           ->addRole('client', 'userRegistered')
           ->addRole('tCompany', 'userRegistered')
           ->addRole('admin', 'userRegistered')
           ->addRole('hotel', 'tCompany')
           ->addRole('company', 'tCompany')
           ->addRole('localAdmin');

$parentsHotelAdmin = array('hotel', 'localAdmin');
$parentsCompanyAdmin = array('company', 'localAdmin');

$this->_acl->addRole(new Zend_Acl_Role('hotelAdmin'),
                     $parentsHotelAdmin)
           ->addRole(new Zend_Acl_Role('companyAdmin'),
                     $parentsCompanyAdmin);
```

Figura. 7.48. Código definiendo los roles en Zend Framework

Finalmente se aplican la configuración acceso de los roles a los recursos. En este paso final se van definiendo reglas que indican los privilegios de cada rol.

```
$this->_acl
          ->allow('guest', 'auth', 'login')
          ->allow('guest', 'index', 'index')
          ->allow('userRegistered', 'auth', 'logout')
          ->allow('userRegistered', 'logout')
          ->allow('userRegistered', 'userResources')
          ->allow('client', 'clientResources')
          ->allow('tCompany', 'tCompanyResources');
```

Figura. 7.49. Código dando los permisos en Zend Framework

Gracias a este control de listas de acceso la barra de navegación que utiliza la API de Zend Framework (Zend_Navigation) es capaz de determinar qué elementos debe mostrar en función del rol que esta autenticado.

Conclusiones Zend_Auth y Zend_Acl

Con la combinación de estos dos elementos que nos ofrece el framework es posible desarrollar un módulo de autenticación y autorización en el sistema. El módulo de autorización se comunica con el módulo de autenticación para comprobar la existencia de una identidad conocida en el sistema, y si es así obtener el rango (rol) al que pertenece.

El potencial de estas herramientas permiten desarrollar aplicaciones web para usuarios con diferentes roles de una manera potente, limpia y segura.

Integración de Google Maps

Google Maps es un servicio gratuito de Google. Constituye un servidor de aplicaciones de mapas en la web y ofrece imágenes de mapas en los que se puede navegar en altura, inclinación, latitud y longitud. Es posible visualizar los mapas dos tipos de mapa principalmente ya que el resto derivan de ellos:

- Roadmap: que dibuja las regiones y vías.
- Satellite: que muestra fotografías satélite de las localizaciones.

Además, las últimas versiones de estos mapas incluyen construcciones en tres dimensiones y el conocido como streetview, que es un entorno virtual en tres dimensiones creado a partir de fotografías reales de las vías de las ciudades.

Imagen 7.1. Captura Google Maps IUCTC

La API de Google Maps permite a las aplicaciones web integrar los mapas de Google. Cuenta con una gran biblioteca de métodos para modificar el mapa con

una gran cantidad de elementos diferentes. En este proyecto las necesidades son las siguientes:

- Mostrar un mapa: Permite la visualización de un mapa tanto a la compañía que registra actividades como al cliente que las visualiza.
- Localizar un punto a través de una dirección: Facilita la obtención del punto exacto de una actividad a partir de la dirección que indica una compañía.
- Obtener las coordenadas de un punto: Permite almacenar en la base de datos del sistema la localización exacta de una actividad.
- Localizar un punto a través de unas coordenadas: Indica al mapa la localización exacta de una actividad con tan sólo dos números, evitando almacenar direcciones.
- Crear un marcador para un punto determinado: Ayuda a que el cliente tenga una clara percepción del lugar de la actividad.

Para cubrir estas necesidades se ha empleado Google Maps Javascript API V3 con licencia gratuita que cuenta con las funciones que se muestran en la siguiente tabla comparativa:

Funciones	API de Google Maps	API de Google Maps for Business
Street View	✓	✓
Servicio web de codificación geográfica	2.500 solicitudes diarias	100.000 solicitudes diarias
Servicio web de rutas	2.500 solicitudes diarias con 10 hitos por solicitud	100.000 solicitudes diarias con 23 hitos por solicitud
Servicio web de matriz de distancia	100 elementos por consulta 100 elementos cada 10 segundos 2.500 elementos diarios	625 elementos por consulta 1.000 elementos cada 10 segundos 100.000 elementos diarios
Servicio web de elevación	2.500 solicitudes diarias con 25.000 muestras diarias	100.000 solicitudes diarias con 1.000.000 muestras diarias
Resolución máxima del API de Google Static Maps	640 x 640	2.048 x 2.048
Escala máxima del API de Google Static Maps	2X	4X
Resolución máxima	640 x 640	2.048 x 2.048

del API de imágenes de Street View		

Tabla 7.5. Funciones de Google Maps

Como se puede observar en la tabla superior la versión Business ofrece mayor respuesta a la demanda de peticiones y resolución pero esto no es algo necesario inicialmente para este proyecto.

La siguiente tabla de métodos de la API Javascript V3 de Google Maps han sido utilizados durante el desarrollo de la aplicación.

Método	Descripción
`Map(mapDiv:Node, opts?:MapOptions)`	Crea un nuevo mapa dentro del contenedor HTML, que normalmente es un elemento DIV.
`setCenter(latlng:LatLng)`	Centra el mapa en unas coordenadas dadas por el conjunto latitud-longitud.
`setZoom(zoom:number)`	Indica al mapa el zoom que debe aplicar.
`setMapTypeId(mapTypeId:MapTypeId|string)`	Elige tipo de mapa que debe mostrarse (terrain, hybrid, roadmap o satellite).
`geocode(request:GeocoderRequest, callback:function(Array.<GeocoderResult>, GeocoderStatus))`	Busca una localización a partir de una dirección pudiendo indicar los límites en coordenadas entre los que buscar, unas coordenadas como punto central de la búsqueda y una región que acote la búsqueda. Devuelve todos los resultados obtenidos para esa dirección incluyendo las coordenadas exactas de cada uno.
`Marker(opts?:MarkerOptions)`	Crea un nuevo marcador con las opciones especificadas. Es posible indicarle una gran cantidad de opciones pero las más básicas son *map*, que indica en que mapa se inserta el marcador; y *position*, que es indicada en coordenadas *x* e *y* posicionando el marcador en el mapa.

Tabla 7.6. Métodos de Google Maps empleados

Control contra CSRF

El CSRF significa en inglés *Cross-site request forgery*, es decir, falsicación de petición en sitios cruzados. Se puede concebir como un tipo de exploit malicioso de un sitio web que acepta comandos no autorizados de un usuario en el que no confía.
Este tipo de ataque utiliza la vulnerabilidad de una aplicación web realizándole una petición que hace una acción como si fuera un usuario autorizado. De esta manera es posible que alguien que no se encuentre ni tan siquiera en la web pueda provocar acciones en las que no debería tener acceso.

Este tipo de ataques funcionan al incluir en un link o script de una página una serie parámetros que el usuario conocer o supone que han sido autenticados.

Veamos cómo transcurre el este ejemplo. Alicia quiere transferir a Carlos 100€ usando banco.com. La petición generada por Alicia es la siguiente:

```
POST http://banco.com/transfer.do HTTP/1.1
...
...
...
Content-Length: 19;

acct=Carlos&amount=100
```

Figura. 7.50. Ejemplo CSRF parte 1

Por otra parte María se da cuenta de que esta aplicación web también podría ejecutar una petición con los siguientes parámetros:

```
GET http://banco.com/transfer.do?acct=Carlos&amount=100
HTTP/1.1
```

Figura. 7.51. Ejemplo CSRF parte 2

Ahora Maria podría aprovechar esta vulnerabilidad contra Alicia enviándole un enlace que haría que se transfiera la cantidad deseada desde Alicia a María:

```
<a
href="http://bank.com/transfer.do?acct=MARIA&amount=100000">¡M
ira mis fotos!</a>
```

Figura. 7.52. Ejemplo CSRF parte 3

Para evitar este tipo de ataques se ha utilizado un validador de Zend Framework (Zend_Validator_Hash) que genera un código hash para cada formulario que muestra al usuario. El servidor al recibir el formulario rellenado debe contener el código hash único para cada usuario con el que fue enviado para que sea validado.

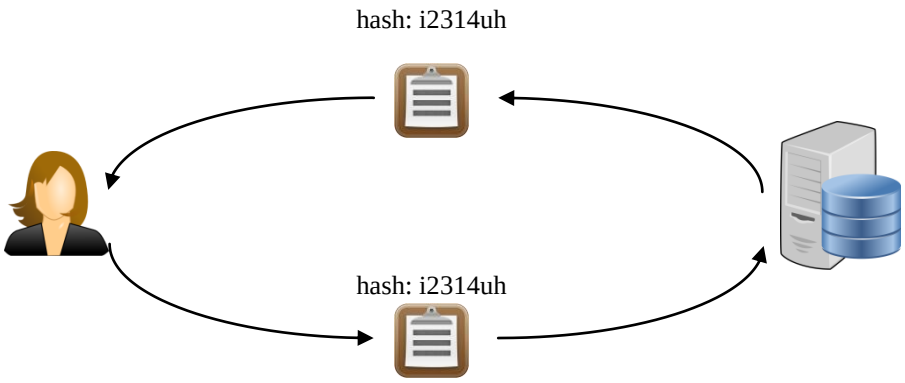


Figura. 7.53. Flujo de Zend Validation Hash

Renombrado de imágenes con hash

En las aplicaciones web que almacenan imágenes cargadas por los usuarios debe haber una política de gestión para manejar los siguientes inconvenientes:

- Tamaño de las imágenes almacenadas: Si el usuario no tiene restricciones puede subir imágenes de cualquier tamaño y estas pueden no ajustarse al formato de la página e incluso sobre cargar el servidor.
- Nombres replicados: Este es el mayor problema ya que un usuario puede tener una imagen con el mismo nombre que otro usuario y ser imágenes diferentes, con o no se dejaría subir una imagen a ese usuario o se borraría la del usuario anterior.

Todos estos problemas han sido solucionados aplicando los métodos habituales que se utilizan en las aplicaciones web:

- Renombrado de la imagen: Al realizar esta transformación se soluciona el problema de no poder subir dos imágenes con el mismo nombre a la aplicación.
- Escalado de la imagen: Todas las imágenes son transformadas al mismo tamaño. De esta manera es posible tener controlado el tamaño que van a ocupar estos recursos.

Procesamiento de las imágenes en el sistema:

Para realizar este proceso se ha optado por la técnica de renombramiento por hash.

1. Se realiza el hash del contenido de la imagen mediante la función resumen md5. Esta función nos devuelve un código de 32 caracteres en hexadecimal, lo que nos da 16^{32} combinaciones diferentes posibles.
2. La imagen se renombra de acuerdo a la categoría que va asociada dentro de la aplicación (company o activity) concatenada con el código hash obtenido anteriomente, quedando algo similar a *activity_5a44c7ba5bbe4ec867233d67e4806848.jp*. Con ello podemos asegurar que no habrá colisión de nombres.
3. Utilizando los métodos para imágenes que nos proporciona la librería de PHP, la imagen es transformada a un tamaño de 260x160 pixeles.

```
$icon = $activityCreateForm->icon->getTransferAdapter();
$iconHash = $icon->getHash('md5');
$iconRename = APPLICATION_PATH .
              '/../public/uploads/imgs/activity_' .
              $iconHash . '.jpg';
$icon->addFilter('Rename', array('target' => $iconRename,
                                 'overwrite'    => true,
                                 )
                 );
// Resize image
list($weight, $height) = getimagesize($iconRename);
$thumb = imagecreatetruecolor(260, 180);
$origin = imagecreatefromjpeg($iconRename);
imagecopyresized($thumb, $origin, 0, 0, 0, 0, 260, 180,
                 $weight, $height);
imagejpeg($thumb, $iconRename);
```

Figura. 7.54. Código para el tratamiento de imágenes en el sistema

Ajax en Zend_Framework

Introducción.

Zend Framework provee al desarrollador con su librería de varias utilidades que facilitarán la integración y el uso de la tecnología AJAX en la aplicación web. La potencia reside en que no es necesario crear acciones nuevas en un controlador para llamadas que necesitan el mismo resultado pero con distinto formato. Esto ocurre cuando visualizamos una actividad mediante el portal web y cuando la visualizamos mediante la aplicación del cliente móvil, se muestra el mismo resultado pero en distinto formato.

ContextSwitch y AjaxContext es como se llaman los elementos que permiten trabajar con esta tecnología cómodamente.

ContextSwitch: Es un action helper para los controlador que intenta facilitar el manejo de los diferentes formatos de respuesta a dar en función de las peticiones.
AjaxContext: Es una versión especializada de ContextSwitch que permite devolver respuestas a XmlHttpRequests.

Las funciones que realiza que realizan son:

- Deshabilitar las plantillas si están habilitadas.
- Definir un sufijo alternador de la vista, dando una vista separada para el contexto seleccionado.
- Enviar la cabecera de la respuesta apropiada para el contexto deseado.
- De manera opcional, realizar llamada a callbacks para realizar la configuración del contexto o un pos procesado.

Configuración en Zend Framework.

Para conseguir que el controlador conozca que tenemos esta característica inicializada hemos de indicárselo en la función de arranque *init()*:

```
public function init()
{
     $ajaxContext = $this->_helper->getHelper('AjaxContext');
     $ajaxContext->addActionContext('view', 'html')
                 ->addActionContext('form', 'html')
                 ->addActionContext('process', 'json')
                 ->initContext();
}
```

Figura. 7.55. Ejemplo de código para AJAX en Zend Framework

Una vez indicado esto, cuando el contexto en AJAX es detectado las acciones *view* y *form* usarán las vitas *view.ajax.phtml* y *form.ajax.phtml* respectivamente. La acción *process* devolverá simplemente los datos en formato JSON.

En el caso de que se conozca que la llamada a una determinada acción debe devolver los datos en formato JSON es posible ayudarse de los helpers y métodos que proporciona la librería de Zend:

Método	Descripción
`$this->_helper ->viewRenderer ->setNoRender(true);`	Obliga a la interfaz a no mostrar la vista.
`$this->_helper->layout ->disableLayout();`	Deshabilita la plantilla.
`Zend_Json::encode($data);`	Codifica los datos en formato JSON.

Tabla 7.7. Métodos para trabajar con AJAX en Zend Framework

Para las llamadas AJAX desde el portal web se ha utilizado el método $.ajax() de la librería JQuery que nos permite realizar estas llamadas asíncronas pudiendo especificar una gran variedad de opciones. En la siguiente tabla se muestran los parámetros más utilizados en la función *jQuery.ajax([settings])*:

Parámtros	Descripción
`data`	Constituyen los datos que van a ser enviados en la petición.
`dataType`	Indica en que formato se esperan los datos de respuesta (xml, json, script o html).
`statusCode`	Representa el código de respuesta (404, etc...).
`success`	Una función que es llamada en caso de el proceso termine correctamente.
`type`	El tipo de petición que se realiza (GET,

	POST, PUT, DELETE).
`url`	El parámetro principal que contiene la url a la que se envía la petición.

Tabla 7.8. Parámetros de la petición AJAX con JQuery

PHPUnit en Zend Framework

PHPUnit es un entorno de para realizar pruebas unitarias en el lenguaje de programación PHP. Fue desarrollado como framework de tests unitarios basado en el framework *JUnit* para Java. Su propósito es que cuanto antes los errores en el código antes podrán ser corregidos. Como cualquier otro framework de pruebas unitarios, PHPUnit utiliza *assertions* para verificar que el comportamiento de una unidad de código es el esperado.

Para testear código con tests unitarios con PHPUnit es necesario seguir las siguientes reglas:

- El nombre de la clase no debe comennzar por "Test".
- El nombre de cada método debe comenzar por "test".

Aplicando estas reglas básicas de la estructura del framework es posible comenzar a construir tests unitarios como el siguiente:

```
Class MoneyTest extends TestCase {

   function test_simpleadd() {
      $money1 = new Money ( 12, "NZD" );
      $money2 = new Money ( 14, "NZD" );
      $expected = new Money ( 26, "NZD" );
      $result = $money1->add( $money2 );
      $this->assertEquals( $expected, $result, "This should pass" );
   }

};

$test = new MoneyTest( "test_simpleAdd" );
$testRunner = new TestRunner();
$testRunner->run( $test );
```

Figura. 7.56. Ejemplo de PHP en Zend Framework

Zend_Test_PHPUnit extiende toda la potencia de PHPUnit a las aplicaciones MVC. Provee de una gran variedad de métodos para realizar tests de la aplicación.

<u>Aserciones incluidas</u>

CSS Selector:

- assertQuery($path, $message)
- assertQueryContentContains($path, $match, $message)

- assertQueryContentRegex($path, $pattern, $message)
- assertQueryCount($path, $count, $message)
- assertQueryCountMin($path, $count, $message)
- assertQueryCountMax($path, $count, $message)

XPath

- assertXpath($path, $message = '')
- assertNotXpath($path, $message = '')
- assertXpathContentContains($path, $match, $message = '')
- assertNotXpathContentContains($path, $match, $message = '')
- assertXpathContentRegex($path, $pattern, $message = '')
- assertNotXpathContentRegex($path, $pattern, $message = '')
- assertXpathCount($path, $count, $message = '')
- assertNotXpathCount($path, $count, $message = '')
- assertXpathCountMin($path, $count, $message = '')
- assertNotXpathCountMax($path, $count, $message = '')

Redirect

- assertRedirect($message = '')
- assertNotRedirect($message = '')
- assertRedirectTo($url, $message = '')
- assertNotRedirectTo($url, $message = '')
- assertRedirectRegex($pattern, $message = '')
- assertNotRedirectRegex($pattern, $message = '')

Response header

- assertResponseCode($code, $message = '')
- assertHeader($header, $message = '')
- assertHeaderContains($header, $match, $message)
- assertHeaderRegex($header, $pattern, $message)

Request

- assertModule($module, $message = '')
- assertController($controller, $message = '')
- assertAction($action, $message = '')
- assertRoute($route, $message = '')

Durante el proyecto para realizar los tests de login en la aplicación se implementó el siguiente código:

```
class UserControllerTest extends
                Zend_Test_PHPUnit_ControllerTestCase
{
    public function setUp()
    {
        $this->bootstrap = array($this, 'appBootstrap');
        parent::setUp();
    }

    public function appBootstrap()
    {
        $this->frontController
             ->registerPlugin(new
Bugapp_Plugin_Initialize('development'));
    }

    public function testCallWithoutActionShouldPullFromIndexAction()
    {
        $this->dispatch('/users');
        $this->assertController('users');
        $this->assertAction('index');
    }

    public function testIndexActionShouldContainLoginForm()
    {
        $this->dispatch('/users/login');
        $this->assertAction('index');
        $this->assertQueryCount('form#loginForm', 1);
    }

    public function testValidLoginShouldGoToProfilePage()
    {
        $this->request->setMethod('POST')
              ->setPost(array(
                  'username' => 'diego',
                  'password' => 'pruebas'
              ));
        $this->dispatch('/users/login');
        $this->assertRedirectTo('/');

        $this->resetRequest()
             ->resetResponse();

        $this->request->setMethod('GET')
             ->setPost(array());
        $this->dispatch('/');
        $this->assertRoute('default');
        $this->assertModule('default');
        $this->assertController('index');
        $this->assertAction('index');
        $this->assertNotRedirect();
    }
}
```

Figura. 7.57. Código para testear el proceso de login con PHPUnit

Implementación de estadísticas y uso Google Charts para su visualización.

Implementación

Para la implementación de estadísticas se han utilizado los siguientes parámetros:

- Número de veces que se muestra una actividad
- Media de las valoraciones de una actividad
- Registro temporal de las visualizaciones de las actividades
- Registro temporal de las uniones a los hoteles (sólo en hoteles)

Su implementación se ha llevado a cabo totalmente bajo peticiones AJAX integrando las respuestas con la API de Google.

Integración con Google Charts

Google Charts es una API de Google gratuita con una gran cantidad métodos y tipos de gráficas con las que poder trabajar.

En este proyecto para ofrecer las estadísticas a las compañías se han utilizado los siguientes tipos de gráficas:

- Pie chart (1)
- Table (2)
- Line chart (3)
- Column chart (4)

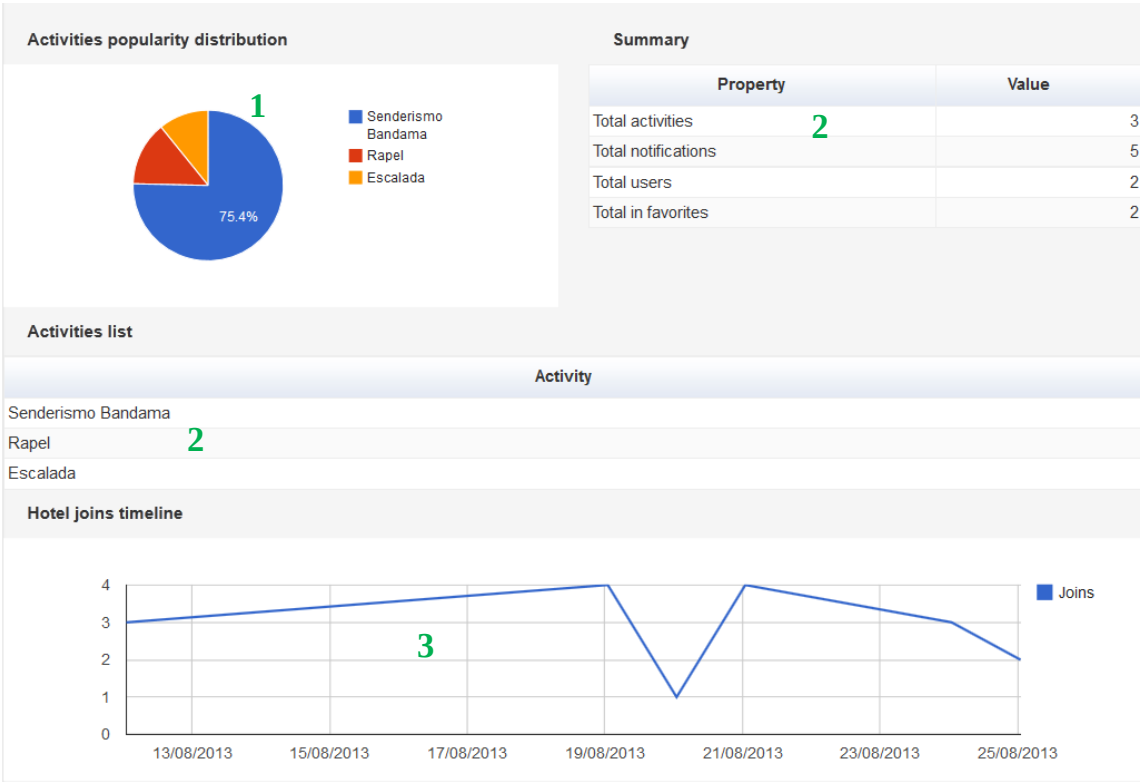


Imagen 7.2. Captura de pantalla de estadísticas de un hotel

Imagen 7.3. Captura de pantalla de estadísticas de una actividad

Las posibilidades para trabajar con los gráficos de Google son amplísimas. Para este proyecto se han utilizado los siguientes métodos:

Método	Descripción
`google.load("visualization", "1", {packages:["corechart"]});`	Carga la librería para el tipo de gráficos "corechart"
`google.setOnLoadCallback(drawActivitiesDistributionChart);`	Ejecuta el callback cuando se carga la página
`new google.visualization.DataTable();`	Crea un nuevo objeto Datatable para poder trabajar con los datos
`<Table>.addRows(<number>);`	Crea <number> filas vacías en la tabla seleccionada
`<Table>.addColumn(<type>,<element>);`	Añade una columna a la tabla del tipo <type> con el elemento <element>
`<Table>.setValue(<row>, <column>, <value>);`	Añade el valor <value> en la fila y columna <row> <column> respectivamente
`new google.visualization.<charttype>(document.getElementById('<bodyelement>'));`	Crea un nuevo objeto gráfico del tipo indicado en el nodo DOM que se le proporciona
`<Chart>.draw(<Table>);`	Dibuja el gráfico con los datos del objeto tabla proporcionados

Tabla 7.9. Métodos de Google Charts utilizados

8.- Módulo de Karotz como interfaz cliente.

8.1.-Estructura de las aplicaciones en Karotz.

Las aplicaciones para Karotz son desarrolladas en el lenguaje JavaScript

Todas las aplicaciones de Karotz siguen la misma estructura, que consiste en un paquete que contiene al menos cuatro ficheros:

- descriptor.xml
- screen.xml
- util.js
- main.js

A ellos, pueden añadirse otros como 'tinyxmldom.js' para trabajar con objetos en xml o cualquier librería necesaria en JavaScript para la aplicación.

A continuación se describen los ficheros básicos de una aplicación para Karotz:

8.1.1.- descriptor.xml.

Este fichero contiene la información necesaria para la publicación y la correcta funcionalidad de la aplicación. En él se define desde el tipo de aplicación hasta los accesos funcionales que puede realizar.

Al ser un fichero xml todas sus propiedades se definen por tags:

<name> : Nombre de la aplicación

<version> : Versión de la aplicación.

<access> : Acceso funcional permitido de la aplicación. Estos son los elementos más importantes ya que amplían o limitan la funcionalidad de nuestra aplicación. Los accesos funcionales definidos para Karotz son:

- led : Debe estar definido si la aplicación utiliza los métodos led.
- ears : Debe estar definido si la aplicación hace que las orejas se muevan.
- button : Debe estar definido si se desea que la aplicación utilice el botón.
- tts : Debe estar definido si la aplicación debe transformar texto a voz.
- webcam : Debe estar definido si la aplicación usa la cámara web.
- asr : Debe estar definido si la aplicación si la aplicación debe reconocer voz
- rfid : Debe estar definido si la aplicación si la aplicación utiliza los nanoztag u otros elementos rfid
- http : Debe estar definido si la aplicación accede a internet, por ejemplo para un servicio web.
- file : Debe estar definido si la aplicación va a realizar lecturas o escrituras en ficheros.
- serial : Debe estar definido si la aplicación utiliza métodos serial

<editor> (Opcional) : Nombre del publicador.
<asrName> (Opcional) : Comando de voz de lanzamiento
<deployment> **(**Opcional) : Para aplicaciones tipo *hosted*, es decir, embebidas en el robot y realizadas en javascript; o *external* si la aplicación funcional mediante la API Rest.
<multiConf> **(**Opcional) : Permite que la aplicación utilice o no varios perfiles y configuraciones, true/false.
<callback> **(**Opcional) : La URL de callback utilizada para aplicaciones basadas en la API Rest.

Un ejemplo de descriptor es el siguiente:

```
<descriptor>
  <name>My first app</name>
  <version>1.0.0</version>
  <accesses>
    <access>http</access>
    <access>tts</access>
    <access>ears</access>
    <access>button</access>
    <access>led</access>
    <access>asr</access>
    <access>multimedia</access>
  </accesses>
  <multiConf>false</multiConf>
  <asrName>top start</asrName>
</descriptor>
```

8.1. Ejemplo de descriptor de fichero de Karotz

8.1.2.- screen.xml.

El fichero screen.xml en las aplicaciones de Karotz representa el panel de control y configuración de la aplicación.

Para este panel se pueden definir los siguientes elementos:

- nanoTrigger: Muestra el panel que gestiona el lanzamiento de la aplicación mediante un nano.
- permanentTrigger: Muestra el panel que permite mantener un *listener* constante para lanzar la aplicación.
- scheduledTrigger: Muestra el panel que permite lanzar la aplicación a una hora determinada.
- scheduledDateTrigger: Muestra el panel que permite lanzar la aplicación a una fecha y hora determinada
- voiceTrigger: Muestra el panel que permite modificar el comando de voz que lanza la aplicación.

Además es posible definir paneles propios utilizando los siguientes elementos:

- Text fields
- Listas desplegables
- Botones

Esto permite que el usuario que instale la aplicación desarrollada pueda modificar opciones de la misma desde el panel de la cuenta de usuario.

La configuración por defecto que se genera en la aplicaciones Karotz es la siguiente:

```
<screen     nanoTrigger="true"
    permanentTrigger="true"
    scheduledTrigger="true"
    scheduledDateTrigger="false"
    voiceTrigger="true">
</screen>
```

Figura. 8.2. Descriptor de los paneles de configuración de la aplicación para Karotz

8.1.3.- util.js.

Este fichero contiene el código necesario para conectar la aplicación a Karotz

```
karotz.connectAndStart = function(host, port, callback, data){
    try{
        karotz.connect(host, port);
      log("connected");
      karotz.start(callback, data);
    }catch(err){
      log(err);
    }
};
```

Figura. 8.3. Fichero de conexión a Karotz

8.1.4.- main.js.

El fichero main.js contiene el código usado en la aplicación. Veamos un ejemplo:

```
include("util.js");

var karotz_ip="localhost"

var buttonListener = function(event) {
    if (event == "DOUBLE") {
        karotz.tts.stop();
        exit();
    }
    return true;
}

var exitFunction = function(event) {
    if((event == "CANCELLED") || (event == "TERMINATED")) {
        exit();
    }
    return true;
}

var onKarotzConnect = function(data) {
    karotz.button.addListener(buttonListener);
    karotz.tts.start("Hello World ! My name is Karotz !", "en",
exitFunction);
}

karotz.connectAndStart(karotz_ip, 9123, onKarotzConnect, {});
```

Figura. 8.4. Ejemplo de código principal para Karotz

8.2.-Módulo cliente para Karotz

La interfaz robótica de Karotz hace que sea demasiado complejo realizar el inicio de sesión de un usuario concreto en EIAT ya que sus elementos RFID son finitos y no programables. Además sus mecanismos de reconocimiento de voz impiden que reconozca una cadena de caracteres que no tenía en su biblioteca a priori, como un nombre de usuario o contraseña.

Por estos motivos el módulo desarrollado utiliza el rol *guest* con el que interactúa de dos formas diferentes: obteniendo las diez primeras actividades y obteniendo una recomendación de una actividad no basada en el usuario.

La aplicación desarrollada para Karotz se ha modularizado en varias funciones que se encargan de realizar todas las tareas necesarias para el correcto funcionamiento de la aplicación. Esta relación de funciones se resume a continuación.

8.2.1.- Funciones

A continuación se detallan las principales funciones desarrolladas para el módulo cliente de Karotz.

getShowmoreFromServer
Obtiene los parámetros detallados de una actividad. Seguidamente los combina con frases para generar un texto que el usuario anónimo pueda entender como el detalle de una actividad.

getShowmore
Comunica al usuario el detalle de la actividad seleccionada y tras ello, ofrece un menú donde volver a repetir el detalle de la actividad o volver al menú principal.

detectActivity
Esta función se encarga de reconocer la actividad seleccionada dentro de la lista ofrecida.

getActivitiesGrammar
Su función es generar una gramática válida para que el sistema ASR de Karotz pueda reconocer la actividad elegida.

getToptenFromServer
Se conecta al servicio web EIAT y obtiene las diez actividades más populares del sistema de recomendación.

getTopten
Interactúa con el usuario y le comunica las diez actividades más populares de la aplicación.

getCategoriesGrammar
Genera la gramática válida para que sistema ASR de Karotz detecte la categoría elegida por el usuario.

getCategoriesFromServer
Obtiene la lista de categorías del servidor EIAT.

selectDayToRecommendation
Se comunica con el usuario para seleccionar el día a realizar la actividad y realizar la petición para la recomendación.

detectCategory
Detecta la categoría seleccionada por el usuario y obtiene su ID para obtener la recomendación.

getRecommendation

Función que se comunica con el usuario para indicarle las categorías disponibles en EIAT y continuar así con el proceso de recomendación.

main
Punto de entrada principal de la aplicación que se comunica con el usuario para conocer si se desea obtener una recomendación o conocer las diez actividades más populares.

8.2.2.- Flujo de opciones del módulo cliente.

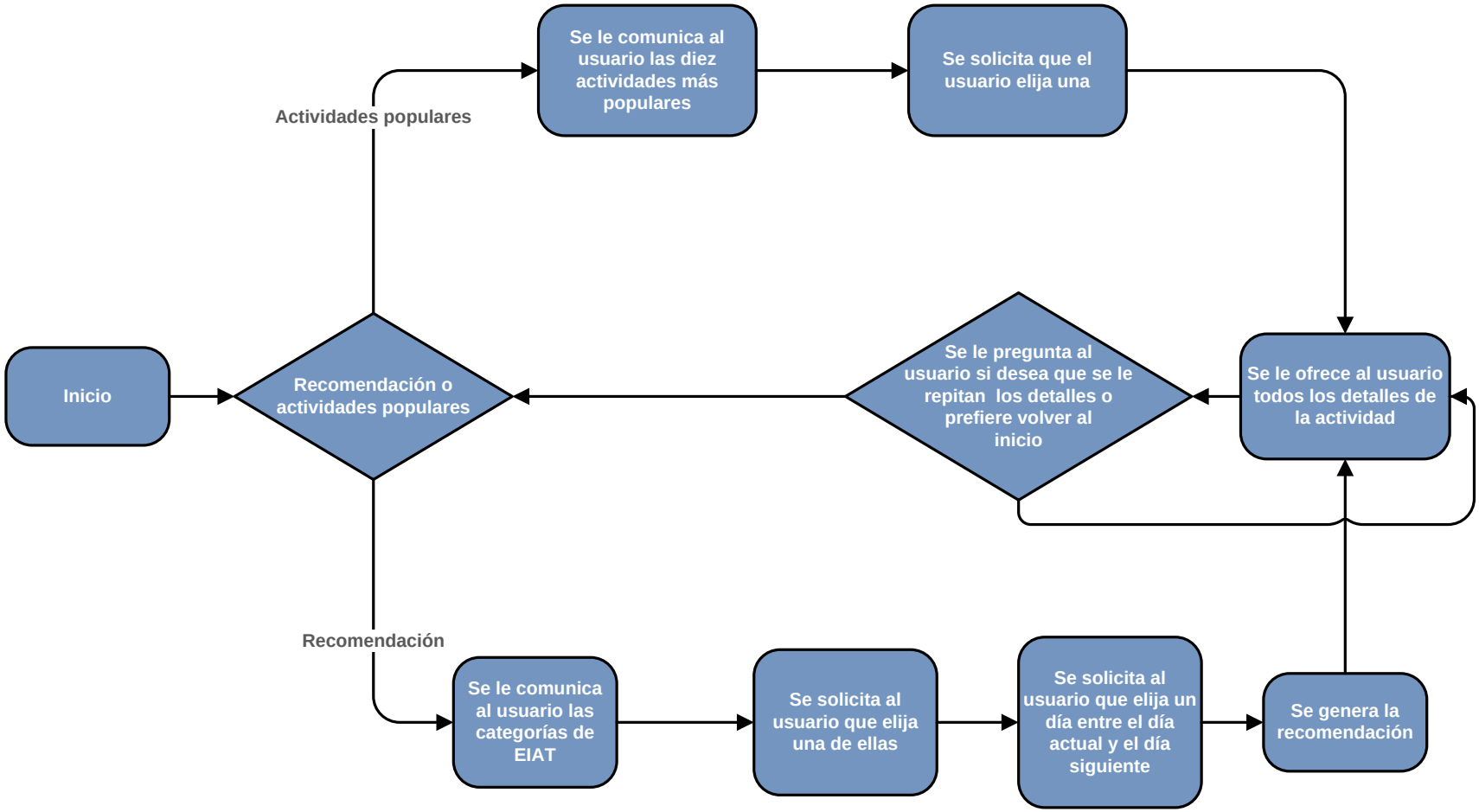


Figura. 8.5. Flujo de opciones en la aplicación para Karotz

9.- Android.

9.1.-Introducción.

Android es un sistema operativo monolítico para dispositivos móviles como smartphones o tablets. Está basado en Linux y soporta arquitecturas ARM, x86, MIPS e IBM POWER. Todas las aplicaciones para este sistema operativo se escriben en lenguaje Java con las librerías de Android. Posteriormente se compilan y se ejecutan en la máquina virtual Dalvik.

9.2.-Estructura de las aplicaciones en Android.

9.2.1.- Fundamentos.

La arquitectura de Android se divide en varias capas con una gran cantidad de módulos:

Figura. 9.1. Arquitectura de Android

- **Aplicaciones:** las aplicaciones base incluyen un cliente de correo electrónico, programa de SMS, calendario, mapas, navegador, contactos y otros. Todas las aplicaciones están escritas en lenguaje de programación Java.
- **Marco de trabajo de aplicaciones:** los desarrolladores tienen acceso completo a las mismos APIs del framework usadas por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades. Este mismo mecanismo permite que los componentes sean reemplazados por el usuario.

- **Bibliotecas:** Android incluye un conjunto de bibliotecas de C/C++ usadas por varios componentes del sistema. Estas características se exponen a los desarrolladores a través del marco de trabajo de aplicaciones de Android; algunas son: System C library, bibliotecas de medios, bibliotecas de gráficos, 3D y SQLite, entre otras.

- **Runtime de Android:** Android incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas base del lenguaje Java. Cada aplicación Android corre su propio proceso, con su propia instancia de la máquina virtual Dalvik. Dalvik ha sido escrito de forma que un dispositivo puede correr múltiples máquinas virtuales de forma eficiente. Dalvik ejecuta archivos en el formato Dalvik Executable (.dex), el cual está optimizado para memoria mínima. La Máquina Virtual está basada en registros y corre clases compiladas por el compilador de Java que han sido transformadas al formato.dex por la herramienta incluida "dx".

- **Núcleo Linux:** Android depende de Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red y modelo de controladores. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila de software.

Cada aplicación tiene un ciclo de vida en la que se encuentra en su propia caja de seguridad:

- Para Android cada aplicación constituye un usuario diferente.
- Por defecto, el sistema asigna a cada aplicación un único *User ID*. De esta forma le da permisos a ese usuario para que gestione sólo él los ficheros que va a utilizar.
- Cada proceso se ejecuta en su propia máquina virtual creando un entorno cerrado para el código de cada una.
- Por defecto, cada aplicación corre en su propio proceso de Linux. Android comienza el proceso cuando la aplicación necesita ser ejecuta y lo finaliza cuando el proceso lo requiere.

El desarrollo en Android también se hace bajo el patrón modelo-vista-controlador.

9.2.2.- Componentes de aplicación.

Las aplicaciones en Android son esencialmente un conjunto de bloques. Cada componente es un punto diferente por donde el sistema puede entrar a la aplicación. Existen cuatro tipos de estos componentes en una aplicación Android:

- **Actividades:** Una actividad representa una pantalla simple en la interfaz de usuario, por ejemplo un listado de emails. Las actividades trabajan de forma conjunta pero cada una se trata de forma separada en la aplicación.
- **Servicios:** Un servicio es un componente que funciona de fondo en el sistema operativo y realiza largas operaciones o tareas en remoto.
- **Proveedores de contenido:** Un proveedor de contenidos gestiona un conjunto compartido de datos de aplicación. Es posible almacenar datos en el sistema a través de una base de datos SQLite.
- **Broadcast receivers:** Es un componente que responde a las llamadas del sistema realizadas en broadcast. La mayoría de estas llamadas se originan desde el sistema. Un ejemplo de esta llamada es que la pantalla se ha apagado, o que la batería se encuentra baja o que se capturó una imagen. Aunque las aplicaciones pueden también inicializar broadcast indicando por ejemplo que un dato ya se ha descargado para que lo use otra aplicación.

9.3.-Desarrollo del módulo cliente del sistema de recomendación de actividades.

9.3.1.- Introducción.

La aplicación que actuará en los dispositivos Android como cliente del sistema de recomendación de actividades tiene las siguientes características y funcionalidades:

- Búsqueda de actividades. Donde se puede parametrizar las búsquedas de forma similar a como se hace en el portal web.
- Recomendación de actividades. Que reporta al usuario una actividad recomendada en función de su perfil y actividad.

- Actividades favoritas. Donde se muestra un listado con las actividades que el usuario ha seleccionado como favoritas.
- Hoteles. Que permite gestionar la búsqueda de hoteles y su unión a ellos. Y además es posible visualizar las notificaciones, actividades propias y recomendadas del hotel.

Esta aplicación se concibe como un cliente, por lo que será el servidor quien gestione todos los datos de las peticiones y respuestas.

9.3.2.- Prototipos de las interfaces de usuario.

Interfaz en la pestaña de búsqueda

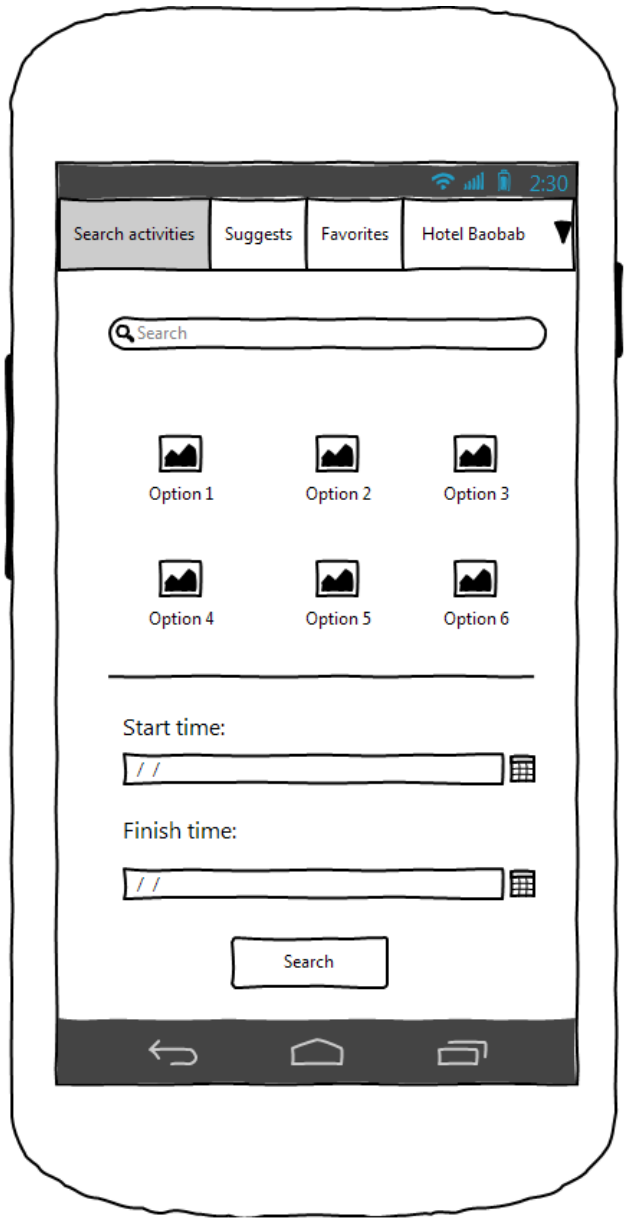


Figura. 9.2. Prototipo de interfaz en Android

Con esta interfaz se pretende que el usuario tenga en la palma de su mano las principales funciones del sistema. En ella podemos ver como se mantienen los campos que aparecen en el portal web para parametrizar las búsquedas.

9.3.3.- Diseño de la arquitectura.

Como se ha ido comentando durante todo el proyecto la aplicación para Android constituye un cliente para la aplicación. El usuario cliente debe autenticarse en la aplicación para acceder a sus funciones.

En la siguiente tabla se definen los controladores y vistas de la aplicación:

Actividad	Descripción
`Activity_login`	Actividad que controla el login del usuario en la aplicación. Es lo primero que se ve cuando se ejecuta.
`Main_activity`	En esta actividad se muestran la interfaz principal de la aplicación, desde donde están disponibles todas las acciones. El controlador se encarga de gestionar toda la lógica de esta actividad, que es la que mayor carga lleva.
`Activity_detail`	Maneja el contenido en detalle de una actividad. Desde esta actividad es posible añadirla o eliminarla de favoritos y localizarla en el mapa.
`Activity_map`	Localiza la actividad en un mapa con sus coordenadas utilizando Google Maps.
`Activity_about`	Interfaz estática que muestra la información de la aplicación.

Tabla 9.1. Actividades en Android

Para que la aplicación adquiera funcionalidad es necesario hacer uso de otras clases que ayudan a completar el funcionamiento de la aplicación. Esas clases se muestran a modo de resumen en la siguiente tabla:

Clases	Descripción
`Activity_adapter`	Clase que sirve como adaptador para poder personalizar el contenido y formato que se muestra en las listas de actividades
`Date_Picker`	Clase que permite la creación de un cuadro de diálogo para que el usuario seleccione el día.
`Time_Picker`	Clase que permite la creación de un cuadro de diálogo para que el usuario seleccione la hora

ListThreeItems	Clase para el manejo de listas de cuatro elementos con tres tipos String y uno tipo Integer
Network	Clase estática que permite mantener la sesión activa del usuario durante todo el ciclo de vida de la aplicación una vez logueado.
Params	Clase que gestiona el almacenamiento de parámetros de la aplicación en una base de datos SQLite

Tabla 9.2. Clases de utilidad en Android

Transferencia de datos

La aplicación cliente de Android se conecta al servidor donde al loguearse crea una sesión que mantiene durante todo el ciclo de vida de ejecución de la aplicación.

A continuación cada vez que el usuario desee realizar cualquier acción que requiera la carga de datos, la aplicación se conecta al servidor y realiza una petición con los datos que el usuario desee recibir. Finalmente, el servidor le envía una respuesta con los datos en formato JSON

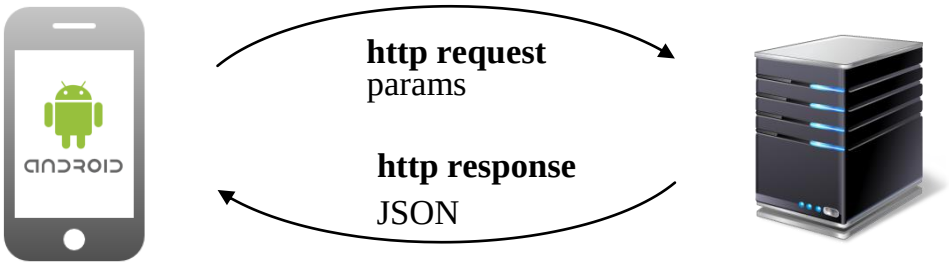


Figura. 9.3. Transferencia de datos entre el cliente y el servidor

Listado y descripción de componentes y elementos utilizados para el desarrollo de la aplicación

- Layout: Para la aplicación se ha trabajado y utilizado la mayoría de elementos de una interfaz para proporcionar al usuario una mejor experiencia. Entre ellos se encuentran:
  - LinearLayout
  - RelativeLayout
  - FrameLayout
  - ViewStub
  - Spinner
  - Button

  - TextView
  - EditText
  - ListView
  - Tabs
  - ImageView
- Listeners: Son métodos que escuchan las interacciones con el elemento de la interfaz y son disparados cuando ocurre el evento adecuado, como por ejemplo el clic de un botón.
- Asynctasks: Las tareas asíncronas permiten dividir la aplicación en varios hilos de forma que el usuario no experimente *lags*. En *background* se va ejecutando la tarea asíncrona mientras que la interfaz sigue estando activa. Este es uno de los elementos más útiles porque permite que no se bloquee la aplicación mientras se cargan datos y son posteriormente lanzados a la interfaz.
- SQLite: Es el almacenamiento nativo de Android. Se gestiona de manera similar a una base de datos convencional y permite almacenar en el dispositivo móvil datos de forma permanente.

10.- Resultados y conclusiones.

10.1.- El robot Karotz. Pros y contras.

El robot Karotz tiene grandes ventajas pero también inconvenientes. Este robot de bajo coste tiene una gran variedad de funcionalidades que se han comentado a lo largo de la memoria de este proyecto pero en la conclusión es necesario dejar patente las limitaciones que también tiene.
Uno de los grandes inconvenientes del robot es la necesidad de estar continuamente conectado a internet para poder estar en funcionamiento. Esta necesidad no guarda relación con que la aplicación que se vaya a ejecutar necesite conexión a internet o no, Karotz siempre la necesita.
Por supuesto el gran punto fuerte del robot es el reconocedor de voz, este elemento permite que el sistema cobre vida y la interacción sea más natural. El reconocedor de voz de Karotz no permite reconocer cadenas que previamente no se le hayan comunicado que va a recibir, esto es, si en la gramática ABNF definida no consta una de las palabras que el robot va a escuchar, la frase no se reconocerá. Por este motivo no es posible realizar tareas como el inicio de sesión en la aplicación y la flexibilidad se reduce. Aunque también es cierto que funciona en entornos algo ruidosos y procesa de forma rápida.
Por otra parte, la documentación para desarrollo de aplicaciones en Karotz no está muy avanzada y no parece que vaya a ir a más de momento.

Sin embargo Karotz, teniendo en mente el coste del robot resulta toda una joya. Ha sido posible que mediante comandos de voz el robot interactúe con el servidor de EIAT y con el usuario de forma transparente, algo que supone un objetivo alcanzado para este proyecto enfocado a las tecnologías en el turismo.

10.2.- Uso de Zend Framework para el desarrollo del servidor.

Una de las primeras cuestiones que uno se plantea cuando quiere desarrollar un sistema como el de este proyecto es por donde empezar. Siguiendo todas las fases de la Ingeniería del Software llega la hora de decidir cómo implementar la aplicación.
Teniendo en cuenta todos los componentes utilizados y la extensión de la aplicación se hizo necesario basarse en una estructura que pudiera ayudar a establecer cada objeto en su sitio y utilizara un flujo de peticiones y respuestas a la aplicación coherentes y no repetitivas. En principio se barajó entre dos alternativas: Zend Framework y Symfony. Symfony por ser un framework del que tenía nociones básicas y Zend Framework porque como bien me recomendó mi tutor Alexis es considerado de los mejores (si no el mejor).

Finalmente el sistema ha sido íntegramente desarrollado bajo la arquitectura de las aplicaciones en Zend Framework v1.12. Este framework tiene una curva de aprendizaje bastante alta pero con el tiempo resulta beneficioso. Una vez que se va conociendo cómo es el flujo de ejecución y el manejo de todos sus componentes, sería posible desarrollar aplicaciones en muy poco tiempo haciendo uso de toda la potencia de su arquitectura y librerías.
Gracias a todos sus componentes ha sido posible tener elementos tan importantes como un sistema de autorización, un sistema de control de listas de acceso (ACL), validación de formularios, interacción coherente y mapeada con la base de datos, paginación, sencillo manejo con respuestas JSON, control de peticiónes y respuestas AJAX, etc.
Sin duda, y no menos importante, el trabajar con el patrón modelo-vista-controlador permite abstraerse e implementar componentes reutilizables.
Otro punto importante a tener en cuenta es que Zend Framework trabaja cien por cien con objetos, lo cual me ha permitido continuar el rodaje y aprendizaje continuo con esta filosofía, que es utilizada de forma profesional.
Por todo ello el uso de un framework y en concreto Zend Framework ha sido muy fructífero y desde luego muy recomendable para desarrollar cualquier aplicación web.

10.3.- Desarrollo de una aplicación con sistema de usuarios.

El desarrollo del sistema de usuarios ha sido uno de los grandes hándicaps de este proyecto. Desde el principio era conocido que era necesario un sistema de usuarios debido a cómo los roles estaban estructurados.
Esta parte del proyecto ha comprendido un largo periodo del proyecto pues afecta a cada una de las acciones que son llevadas a cabo en el sistema. Para ello inicialmente se buscaron varias alternativas sobre cómo crear un sistema de usuarios para una aplicación web y finalmente la alternativa elegida fue un componente de Zend Framework.
Zend Auth es el componente del que hablamos y su diseño basado en el patrón Singleton de Ingeniería del Software hace posible gestionar de forma correcta a cada usuario en el servidor. Además este sistema era compatible con la forma de gestionar los usuarios en la aplicación desarrollada para Android, algo que había que tener en cuenta.

Cómo se quiso implementar un sistema seguro de usuarios con control de listas de acceso fue necesario aprender ciertas técnicas de cifrado criptográfico que se utilizan de forma profesional hoy en día.

La implementación ha sido costosa en tiempo pero excelente en resultados. Ha sido probado con tests en PHPUnit y ha superado todos con éxito.

10.4.- Sistema de recomendación de actividades.

Los usuarios en general desean obtener algo adecuado a sus gustos y al momento de forma sencilla. Este es el motivo básico por el que existen los sistemas de recomendación. Amazon es una veterana en ellos y en este proyecto se quería profundizar, experimentar e implementar uno, ya que el enfoque de la aplicación es similar al de una tienda de Amazon.
Para ello se ha leído numerosa bibliografía y documentación. Existen ya hoy en día varios modelos de sistemas de recomendación pero aún es un campo abierto. En este proyecto se han seguido las técnicas de los sistemas de recomendación implementando un sistema de recomendación final que recomienda a cada usuario una actividad basada en los que se conoce de él y sus vecinos más próximos.

Definitivamente este es uno de los puntos fuertes del proyecto, ya que lo que se pretendía es que un usuario que buscaba actividades recibiera una actividad perfecta para él basada en sus gustos. Los resultados obtenidos tras las pruebas son totalmente coherentes y satisfactorios.

10.5.- Desarrollo de aplicaciones en Android.

El desarrollo extra de una aplicación en Android suponía todo un reto. Aunque previamente se había hecho prototipos para conocer el alcance que podía tomar el proyecto no se conocía con certeza el coste temporal de lo que se pretendía ni a hasta dónde se iba a llegar.
Finalmente su desarrollo ha sido terminado con una funcionalidad y aspecto más que aceptables. Se ha conseguido que la aplicación en Android posea prácticamente las mismas funciones que la parte de un cliente web y todo ello con un buen rendimiento.
La conclusión final de desarrollar la aplicación en Android es que me ha sido posible profundizar más en el lenguaje Java y conocer y valorar la arquitectura y el desarrollo de aplicaciones para Android.

10.6.- Uso de componentes, tecnologías y estrategias.

En este apartado de las conclusiones me gustaría remarcar la importancia de la reutilización e integración de código. Gracias a librerías y componentes como JQuery, JQuery UI, Bootstap, Spinner y otras ha sido posible mostrar un buen aspecto y funcionalidad. Resulta imprescindible y recomendable en este tipo de proyectos utilizar este tipo de elementos que colaboran de forma activa al desarrollo de las aplicaciones.

Git. El control de versiones, tanto para el desarrollo del servidor como para la aplicación para Android ha supuesto un componente esencial para el avance del proyecto. Con este proyecto he conseguido comprender mejor y coger soltura

con la utilización de estos elementos que ahora me resultan imprescindibles para desarrollar cualquier aplicación de tamaño y duración razonable.

PHPUnit. Desde segundo de carrera nos enseñaron a realizar tests unitarios en las aplicaciones aunque la realidad es que no siempre los realizamos. En este proyecto se ha utilizado PHPUnit (muy similar a JUnit para Java) para testear el correcto funcionamiento del login de la aplicación. Esto ha permitido conocer esta herramienta de cerca y saber aplicarla para otros casos como JUnit para Android.

10.7.- Resultado final del proyecto.

La conclusión final de este proyecto es que ha sido un proyecto largo pero en el que he adquirido una gran cantidad de conocimientos en áreas que ya conocía y también en disciplinas nuevas. Ha sido un proyecto que ha finalizado cumpliendo los objetivos y metas que se propusieron teniendo como resultado una aplicación web como servidor y servicio web, una aplicación cliente para Karotz y una aplicación cliente para Android. Todo ello conforma un completo sistema de recomendación de actividades ofrecido tanto como sugerencias como escaparate para el turista.

Esto permite que la salida de este proyecto concluya con un sistema real donde en cada habitación de un hotel pueda haber un Karotz y cada huésped tenga en su móvil instalada la aplicación cliente del sistema de recomendación de actividades.

11.- Trabajo futuro.

Sin duda, este proyecto pertenece a ese grupo de aplicaciones que pueden llegar a convertirse en algo más. Su expansión se puede abrir por una gran cantidad de frentes, algunos de ellos se comentan en los siguientes apartados.

11.1.- Ampliación de los servicios del sistema.

Podemos encontrarnos información por todos lados pero no organizada. El turista continuamente se encuentra en busca de información y necesita que le proporcionen sólo la parte que le interesa.

11.1.1.- Servicios meteorológicos.

La adición de los servicios meteorológicos en el sistema es una posible mejora de la aplicación. Para el turista este tipo de información es muy relevante, ya que en función de ella puede planificar qué actividades hará en el futuro.
Además, en combinación con el sistema de recomendación puede hacer que las sugerencias a los usuarios sean aún más lógicas teniendo en cuenta este importante factor.

11.1.2.- Servicios de información de vuelos.

Otro elemento importante puede ser conectar la aplicación con algún servicio web para proporcionar al usuario las últimas novedades de los vuelos. Con ello los usuarios pueden conocer antes de abandonar el hotel si su vuelo ha sido retrasado o continúa en hora.

11.1.3.- Reporte de estadísticas más completo.

Todas las empresas desean mejorar productos y conocer cuáles son los más utilizados para analizarlos. Por ello, el ofrecerle cada vez estadísticas más completas de sus actividades en el sistema les permitiría que hicieran ese análisis y trataran de integrarse cada vez más dentro de la aplicación.
Cuando un sistema resulta útil no sólo por su propaganda sino por su *feedback* aumenta su valor.

11.1.4.- Capacidad para realizar reservas de actividades directamente desde la web.

No con todas las actividades es posible o necesario reservarlas. En cambio para otras como el alquiler de una cancha de tenis o las entradas para un espectáculo sí puede ser muy útil. Por eso integrar este servicio dentro de la aplicación aumentaría su utilidad y uso para los usuarios.

11.1.5.- Geoposicionamiento y uso del servicio de rutas de Google Maps.

En el proyecto se decidió integrar los mapas de Google para posicionar las actividades, pero si además a los usuarios les proporcionamos la ruta para llegar desde la misma aplicación no decaerán en ir o buscar cómo llegar de otra manera. Utilizando el gps del móvil o la tableta y con la API de Google Maps se podrían trazar las rutas para llegar a las actividades.

11.1.6.- Interfaz multi-idioma.

Hoy en día es algo básico para cualquier aplicación pero cuando se desarrollan aplicaciones orientadas al turismo resulta imprescindible.
Teniendo en cuenta que en Gran Canaria o Canarias la mayor parte de turismo es inglés y alemán tener la aplicación en estos idiomas es elemental.
Zend Framework provee de un gestor de idiomas (Zend_Translate) que permite abstraer la lógica de la aplicación del idioma de la interfaz.
Por otra parte, el sistema operativo y la API de Karotz permite operar en cuatro idiomas: español, inglés, francés y alemán. De modo que se puede aprovechar esta característica para que el módulo cliente del robot soportara la comunicación de cualquiera de estas cuatro lenguas.

11.2.- Búsqueda de alternativas o complementarias a Karotz.

Como ya se ha comentado en las conclusiones, el robot Karotz tiene todas las características y funcionalidades para ser el cliente ideal del sistema pero la necesidad de mantenerse conectado no sólo a internet sino concretamente al servidor de Aldebaran durante su funcionamiento, hace que su uso sea más que cuestionable.
Encontrar una plataforma similar a Karotz, con la funcionalidad básica de internet y de interfaz de comunicación por voz, puede ahorrar muchos problemas en el futuro.
Existen una gran cantidad de robots zoomórficos que se ajustan a este perfil, aunque también es cierto que elevan su coste. Por otra parte nos encontramos con dispositivos electrónicos de voz y multimedia, como si de un gps o manos libres se tratase, y aunque no presente un aspecto tan amigable también cumplen los requisitos.
Un ejemplo recién nacido es el nuevo robot Aisoy1, de una empresa española, con un coste algo más elevado que Karotz, 200€, pero aún así muy por debajo de la media. Este robot es también programable y está basado en las nuevas tecnologías de desarrollos de robots, Raspberry Pi y ROS, algo que permitiría desarrollar aplicaciones muy completas. Sin duda el futuro de esto proyecto podría estar unido al de este robot.

11.3.- Mejora del sistema de recomendación.

Creo profundamente que este es un punto clave de la aplicación. El ofrecer al usuario lo que desea de manera más precisa asegurará el éxito del sistema. En el proyecto el sistema de recomendación es básico y puede mejorarse teniendo en

cuenta muchos más factores como la climatología, número de personas interesadas ese día en esa actividad o frecuencia de reserva de actividades.

11.4.- Desarrollo de una aplicación cliente para iOS.

En el mundo de los smartphones existen esencialmente dos reinos de sistemas operativos Android y iOS. La implementación del cliente para iOS permitiría aumentar el alcance de la aplicación. La estructura de la aplicación sería muy similar y la implementación seguiría un desarrollo muy parecido a la de Android.

12.- Bibliografía.

Segaran, Toby. "Programming Collective Intelligence Building Smart Web 2.0 Applications". O'Reilly Media.

E. Williams, Hugh; Lane, David. "Web Database Applications with PHP and MySQL, Building Effective Database-Driven Web Sites". O'Reilly Media.

Russell, Stuart; Norvig, Peter . "Artificial Intelligence: A Modern Approach". Pearson.

"PHP Manual".
http://www.php.net/manual/en/

"Zend Framework & MVC Introduction".
http://framework.zend.com/manual/1.12/en/learning.quickstart.intro.html

"Zend Framework Reference".
http://framework.zend.com/manual/1.12/en/reference.html

"Zend Framework".
http://angelorum.blogspot.com.es/2010/09/zend-framework-1-instalacion.html

"PHP CSRF Guard".
https://www.owasp.org/index.php/PHP_CSRF_Guard

"ZFdes".
http://manual.zfdes.com/es/index.html

"Zend Framework API Documentation".
http://framework.zend.com/apidoc/1.12/

"Zend Framework Documentation".
http://www.zendframeworkdocumentation.com/

"Zend Framework Basic Tutorial".
http://www.zendframeworkdocumentation.com/

"Zend Framework and Doctrine ORM".
http://inchoo.net/tools-frameworks/zend/doctrine-orm-and-zend-framework-sample-project-to-get-you-started/

"Using Doctrine 2 in Zend Framework".
http://www.jasongrimes.org/2012/01/using-doctrine-2-in-zend-framework-2/

"Creating a PHP REST API Using the Zend Framework".
http://www.chrisdanielson.com/2009/09/02/creating-a-php-rest-api-using-the-zend-framework/

"Create RESTful Applications Using The Zend Framework".
http://www.techchorus.net/create-restful-applications-using-zend-framework

"Writing a REST Web Service and Client With Zend_Controller".
http://www.zendcasts.com/writing-a-restful-web-service-and-client-with-zend_controller-and-zend_httpclient/2009/04/

"Authorization with Zend_Acl in Zend Framework".
http://www.packtpub.com/article/authorization-with-zend_acl-zend-framework-1.8

"Zend Acl with Authentication and Reflection".
http://www.youtube.com/watch?v=PALEC6XqCZc

"Zend_Acl: autorización y permisos en Zend Framework".
http://otroblogmas.com/zend_acl-autorizacion-y-permisos-en-zend-framework/

"Understanding Zend Framework's Plugin Parts 1 & 2".
http://devzone.zend.com/2508/understanding-zend-frameworks-plugin-parts-1-2/

"View Helpers in Zend Framework".
http://devzone.zend.com/1235/view-helpers-in-zend-framework/

"Performance Optimisation For Zend Framework Applications".
http://www.survivethedeepend.com/zendframeworkbook/en/1.0/performance.optimisation.for.zend.framework.applications#

"Documentación para desarrolladores de Google Maps".
https://developers.google.com/maps/documentation/?hl=es

"Google Charts".
https://developers.google.com/chart/?hl=es

"jQuery API Documentation".
http://api.jquery.com/

"jQuery UI".
http://jqueryui.com/

"Bootstrap".
http://getbootstrap.com/2.3.2/index.html

"MySQL Documentation: MySQL Reference Manuals".
http://dev.mysql.com/doc/

"Android Developer. API Guides".
http://developer.android.com/guide/components/index.html

Salvador Gómez, Oliver. "Curso de Programación Android".
http://www.sgoliver.net/blog/?page_id=3011

"Personalización y Sistemas de Recomendación".
http://sinbad2.ujaen.es/cod/archivosPublicos/presentaciones/sistemasDeRecomendacion.pdf

"Sistemas de recomendación".
http://arantxa.ii.uam.es/~castells/docencia/ir/apuntes/7-recomendacion.pdf

Schafer, Ben; Frankowski, Dan; Herlocker, Jon; Sen, Shilad. "Collaborative Filtering Recommender Systems".
http://www.eui.upm.es/~jbobi/jbobi/PapersRS/CollaborativeFilteringRecommenderSystems.pdf

"Karotz API Javascript".
http://dev.karotz.com/sdk/en/index.htm

"Karotz Manual".
http://karotz.s3.amazonaws.com/doc/Booklet-US%26UK.pdf

13.- Anexos.

Anexo A. Detalle de casos de uso.

Nombre	**Ver detalle actividad**	**ID**	01
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	
Actor principal			
• Usuario			
Personal involucrado o intereses			
• El usuario desea ver una actividad en detalle.			
Descripción			
El usuario visualizará en detalle la actividad, pudiendo ver por ejemplo, los horarios, fechas, precios, fotos y empresa a la que pertenece.			
Trigger			
Selecciona una de las actividades listadas.			
Precondición			
• Haber listado actividades.			
Postcondición			
• El usuario visualiza todos los detalles de la actividad seleccionada. • La actividad actualiza el número de visualizaciones en detalle realizadas.			
Flujo Normal			
1. El usuario selecciona una actividad de la lista. 2. El usuario visualiza la actividad seleccionada en detalle.			
Flujo Alternativo			
Excepción			
1. El sistema reporta indica que no ha sido posible mostrar la actividad seleccionada. 2. Ir al paso 1.			
Includes			
Requisitos Especiales			
Notas			

Nombre	**Ver las top diez actividades**	**ID**	02
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario

Personal involucrado o intereses

- El usuario desea ver la lista de las diez actividades más populares

Descripción

El usuario visualizará una lista que son las diez actividades más populares.

Trigger

Pulsa sobre el botón **Top 10 activities** del módulo de búsqueda.

Precondición

- El módulo de búsqueda debe estar visible

Postcondición

- El usuario visualiza una lista que son las actividades más populares.

Flujo Normal

1. El usuario hace clic sobre el botón **Top 10 activities** del módulo de búsqueda.
2. El usuario visualiza la lista de activiades.

Flujo Alternativo

Excepción

1. El sistema reporta indica que no ha sido posible mostrar la lista de actividades.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Buscar actividades**	**ID**	03
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario

Personal involucrado o intereses

- El usuario desea buscar actividades de empresas en el sistema.

Descripción

El usuario realizará una búsqueda en el sistema en función de unas palabras clave, categorías y otros filtros que mostrarán una lista de actividades que concuerdan con los parámetros.

Trigger

Introducir palabras clave para la búsqueda y otros parámtros y hacer clic en **Search**.

Precondición

- El módulo de búsqueda debe estar visible.

Postcondición

- Se muestran una lista de actividades que concuerdan con los parámetros de la búsqueda.

Flujo Normal

1. El usuario introduce unas palabras clave en el buscador y otros filtros.
2. El usuario hace clic sobre el botón **Search**.
3. Se muestran una lista de actividades que coinciden con los parámetros de búsqueda.

Flujo Alternativo

1. Tras haber realizado la primera búsqueda, se selecciona algún filtro.
2. Al usuario se le muestra una nueva lista de acuerdo a los cambios en los filtros.

Excepción

1. El sistema reporta indica que no hay actividades que coincidan con la búsqueda.
2. Al usuario se le sugiere que pruebe de nuevo introduciendo otros parámetros más generales.

1. El sistema indica que no es posible mostrar resultado de búsqueda por fallo de conexión al servidor y otro motivo relacionado.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Cerrar sesión**	**ID**	04
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario registrado

Personal involucrado o intereses

- El usuario desea cerrar la sesión y cambiar a usuario anónimo.

Descripción

El usuario cerrará la sesión que en esos momentos mantiene con el sistema.

Trigger

Hacer clic sobre el botón **Logout**.

Precondición

Postcondición

- El usuario ya no aparece como logueado en el sistema.

Flujo Normal

1. El usuario hace clic sobre el menú de usuario.
2. El usuario hace clic sobre el botón **Logout**.
3. El usuario cierra la sesión con el sistema y pasa a ser anónimo.

Flujo Alternativo

Excepción

1. El sistema reporta indica que no ha sido posible cerrar la sesión.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Visualizar datos de usuario**	**ID**	05
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario registrado

Personal involucrado o intereses

- El usuario desea visualizar los datos con los que ha quedado registrado en el sistema.

Descripción

El usuario visualizará sus datos de registro y podrá acceder a las funciones de modificación de contraseña.

Trigger

Hacer clic en el botón **Profile**.

Precondición

Postcondición

- El usuario visualiza una pantalla donde se muestran los datos con los que el sistema le tiene registrado.

Flujo Normal

1. El usuario hace clic sobre el menú de usuario.
2. El usuario hace clic sobre el botón **Profile**.
3. El sistema le muestra una pantalla donde aparecen los datos con los que está registrado.

Flujo Alternativo

Excepción

1. El sistema reporta un fallo de que no es posible mostrar los datos del usuario.
2. Ir al paso 1

Includes

Requisitos Especiales

Notas

Nombre	**Modificar datos de usuario**	**ID**	06
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario registrado

Personal involucrado o intereses

- El usuario desea modificar los datos con los que ha quedado registrado en el sistema.

Descripción

El usuario modificará sus datos de registro.

Trigger

Hacer clic en el botón **Modify**.

Precondición

- El usuario debe haber visualizado los datos.

Postcondición

- Se guardan en el sistema los cambios en los datos realizados por el usuario.

Flujo Normal

1. El usuario modifica los datos que esta visualizando
2. El usuario hace clic sobre el botón **Modify**.
3. El usuario es notificado indicándole que los cambios se llevaron a cabo.
4. Los datos quedan modificados en el sistema y el usuario los visualiza.

Flujo Alternativo

Excepción

1. El sistema reporta un fallo de que no es posible modificar los datos del usuario.
2. Ir al paso 1

Includes

Requisitos Especiales

Notas

Nombre	**Ver preferencias**	**ID**	07
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente que desee ver las preferencias que anteriormente introdujo en el sistema.

Descripción

El usuario cliente visualizará las preferencias sobre las actividades que introdujo anteriormente en el sistema.

Trigger

Hacer clic en el botón **Preferences** dentro del menú de usuario.

Precondición

Postcondición

Flujo Normal

1. Hacer clic sobre el menú de usuario.
2. Hacer clic en **Preferences**.
3. Se muestran las preferencias.

Flujo Alternativo

Excepción

Includes

Requisitos Especiales

Notas

Nombre	Modificar preferencias	ID	08
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente quiere modificar las preferencias que anteriormente introdujo en el sistema.

Descripción

El usuario cliente modificará las preferencias sobre las actividades que introdujo anteriormente en el sistema. Para ello cuando se muestran las preferencias aparecen editables.

Trigger

Precondición

- Haber visualizado las preferencias del usuario cliente.

Postcondición

- Los cambios quedan registrados en la base de datos del sistema.

Flujo Normal

1. El usuario cliente se encuentra visualizando las preferencias.
2. El usuario cliente modifica las preferencias seleccionándolas o deseleccionándolas pinchando sobre cada una de ellas.

Flujo Alternativo

Excepción

1. El sistema reporta un fallo al guardar los datos.
2. Volver al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Ver actividades favoritas**	**ID**	09
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente desea visualizar las actividades favoritas que ha añadido anteriormente.

Descripción

El usuario cliente visualizará una lista de actividades que anteriormente ha decidido añadir en su lista de favoritas.

Trigger

Hacer clic en **Favourites** en el entorno del usuario.

Precondición

Postcondición

- Se muestra la lista de las actividades que el usuario anteriormente marcó como favoritas.

Flujo Normal

1. El usuario cliente visualiza sus actividades favoritas pulsando sobre **Favourites**.
2. Al usuario se le muestran la lista de sus actividades favoritas.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible mostrar la lista de actividades favoritas.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Añadir a actividades favoritas**	**ID**	10
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente desea añadir una actividad a su lista de favoritas.

Descripción

El usuario cliente visualizará se encuentra visualizando una lista de actividades y desea añadirla a su lista de actividades favoritas.

Trigger

Hacer clic en **Add to favourites** sobre la actividad en cuestión.

Precondición

- Encontrarse sobre una lista de actividades.
- Haber mostrado el detalle de la actividad.
- La actividad no se encuentra ya en favoritos.

Postcondición

- Se añade la actividad seleccionada a su lista de actividades favoritas.

Flujo Normal

1. El usuario cliente muestra el detalle de una actividad y la añade a su lista de favoritas pulsando sobre **Add to favourites**.
2. Al usuario se le notifica que se ha añadido.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible añadir la actividad a favoritas.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Eliminar actividad favorita**	**ID**	11
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente desea eliminar una actividad de su lista de favoritas.

Descripción

El usuario cliente se encuentra visualizando su lista de actividades favoritas y desea eliminar una de ellas.

Trigger

Hacer clic en **Unfavourite** sobre la actividad en cuestión.

Precondición

- Encontrarse sobre su lista de actividades favoritas.
- Haber mostrado el detalle de la actividad.
- La actividad debe encontrarse en favoritas.

Postcondición

- Se elimina la actividad seleccionada de su lista de actividades favoritas.

Flujo Normal

1. El usuario cliente selecciona una actividad y la elimina de su lista de favoritas pulsando sobre **Unfavourite**.
2. Al usuario se le notifica que se ha eliminado.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible eliminar la actividad de favoritas.
2. Ir al paso1.

Includes

Requisitos Especiales

Notas

Nombre	Ver actividad recomendada	ID	12
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente desea visualizar la actividad que sistema le recomienda.

Descripción

El usuario cliente accede al sistema o va a su panel inicial para ver la actividad recomendada.

Trigger

Pulsar sobre el botón de inicio o loguearse en el sistema

Precondición

Postcondición

- Se muestra la actividad recomendada en detalle.

Flujo Normal

1. El usuario hace clic en el botón de inicio

Flujo Alternativo

1. El usuario se loguea

Excepción

1. El sistema no muestra una actividad recomendada porque no tiene suficientes datos del usuario.
2. El usuario debe seleccionar algunas preferencias.

Includes

Requisitos Especiales

Notas

Nombre	**Ver actividades sugeridas por el hotel**	**ID**	13
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente desea visualizar las actividades que el hotel le recomienda.

Descripción

El usuario cliente visualizará una lista de actividades que son las que el hotel le recomienda.

Trigger

Hacer clic en **Hotel suggestions** en el entorno.

Precondición

- Encontrarse unido a algún hotel.

Postcondición

- Se visualiza una lista de actividades que son las el hotel le recomienda.

Flujo Normal

1. Hacer clic sobre el menú del hotel.
2. El usuario visualiza las actividades que el hotel le recomienda pulsando sobre **Hotel suggestions**.
3. Al usuario se le muestran las actividades.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible mostrar las actividades sugeridas por el hotel.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Ver notificaciones del hotel**	**ID**	14
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente desea visualizar las notificaciones que el hotel al que está unido ha publicado.

Descripción

El usuario cliente visualizará una lista de mensajes de texto, que serán las publicaciones que el hotel haya hecho.

Trigger

Hacer clic en **Notifications** en el menú del hotel.

Precondición

- Estar unido a algún hotel.

Postcondición

- Se muestra una lista de notificaciones para el usuario.

Flujo Normal

1. Hacer clic sobre el menú del hotel.
2. El usuario cliente visualiza las notificaciones del hotel pulsando sobre **Notifications**.
3. Al usuario se le muestran una lista de notificaciones.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible mostrar la lista de notificaciones.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Ver actividades del hotel**	**ID**	15
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente desea visualizar las actividades del hotel.

Descripción

El usuario cliente visualizará una lista de actividades que son las del hotel.

Trigger

Hacer clic en **Hotel activities** en el entorno.

Precondición

- Encontrarse unido a algún hotel.

Postcondición

- Se visualiza una lista de actividades que son las del hotel.

Flujo Normal

1. Hacer clic en el menú del hotel.
2. El usuario visualiza las actividades del hotel pulsando sobre **Hotel activities**.
3. Al usuario se le muestran las actividades.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible mostrar las actividades del hotel.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Vincularse a hotel**	**ID**	16
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente que desea ver las actividades del hotel seleccionado, sus noticias y las actividades que ellos recomiendan.

Descripción

El usuario cliente podrá establecer un vínculo con el hotel que seleccione. De esta forma el usuario cliente podrá ver las noticias o mensajes del hotel, sus actividades y las actividades que el hotel desea ofrecer de terceros.

Trigger

Hacer clic en el botón **Join**.

Precondición

- No estar vinculado a ningún hotel.

Postcondición

- El usuario cliente puede ver las noticias o mensajes, sus actividades y las actividades que el hotel recomienda del hotel seleccionado.

Flujo Normal

1. El usuario cliente busca un hotel mediante el buscador de hoteles.
2. El usuario cliente hace clic sobre el botón **Join**.
3. El usuario cliente selecciona de una lista el hotel al que desea vincularse.
4. El usuario recibe un mensaje indicándole que se ha vinculado al hotel.

Flujo Alternativo

Excepción

1. El sistema reporta un fallo de que no es posible mostrar los hoteles y vuelve a la pantalla principal del usuario.
2. Ir al paso 1

1. El sistema reporta un fallo de que no es posible vincularse al hotel seleccionado.
2. Ir al paso 1

Includes

Requisitos Especiales

Notas

Nombre	**Desvincularse del hotel**	**ID**	17
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario cliente

Personal involucrado o intereses

- El usuario cliente que desea no ver las actividades del hotel seleccionado, sus noticias ni las actividades que ellos recomiendan.

Descripción

El usuario cliente eliminará el vínculo con el hotel. De esta forma el usuario no verá las noticias o mensajes del hotel, sus actividades ni las actividades que el hotel desea ofrecer de terceros.

Trigger

Hacer clic en el botón **Quit** del menú del hotel al que se encuentra vinculado.

Precondición

- Estar vinculado a algún hotel.

Postcondición

- El usuario cliente ya no puede ver las noticias o mensajes, sus actividades ni las actividades que el hotel recomienda del hotel seleccionado.

Flujo Normal

1. El usuario cliente hace clic sobre el menú del hotel al que se encuentra vinculado.
2. El usuario cliente hace clic sobre el botón **Quit**.
3. El usuario recibe un mensaje indicándole que se ha desvinculado del hotel.

Flujo Alternativo

Excepción

1. El sistema reporta un fallo de que no es posible desvincularse al hotel seleccionado.
2. Ir al paso 1

Includes

Requisitos Especiales

Notas

Nombre	Añadir actividad	ID	18
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario tipo empresa (del que derivan hotel y empresa).

Personal involucrado o intereses

- El usuario tipo empresa desea añadir una actividad para que quede registrada en el sistema.

Descripción

El usuario tipo empresa quiere añadir una actividad para que todos los usuarios pueden visualizarla.

Trigger

Hacer clic en **Activity** en el menú de acciones **New**.

Precondición

Postcondición

- La nueva actividad registrada en el sistema queda visible a los usuarios.

Flujo Normal

1. El usuario tipo empresa añade una actividad en el sistema pulsando sobre **Activity** en el menú de acciones **New**.
2. Rellena los campos.
3. Guarda los cambios.
4. Al usuario se le indica que se añadió la actividad.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible añadir una actividad o guardar los cambios.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Ver mis actividades**	**ID**	19
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario tipo empresa (del que derivan hotel y empresa).

Personal involucrado o intereses

- El usuario tipo empresa desea visualizar las actividades que ha añadido anteriormente en el sistema.

Descripción

El usuario tipo empresa desea ver las actividades que añadió anteriormente, de modo que se le muestra una lista de sus actividades para que las visualice.

Trigger

Hacer clic en **Activities** en el panel principal.

Precondición

Postcondición

- Al usuario se le muestra su lista de actividades.

Flujo Normal

1. El usuario tipo empresa visualiza sus actividades pulsando sobre **Activities** del panel principal.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible ver sus actividades.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	Borrar actividad	ID	20
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario tipo empresa (del que derivan hotel y empresa).

Personal involucrado o intereses

- El usuario tipo empresa desea borrar una actividad que anteriormente añadió en el sistema.

Descripción

El usuario tipo empresa desea borrar una actividad suya del sistema para que no aparezca, y que añadió anteriormente.

Trigger

Hacer clic en **X** sobre la actividad.

Precondición

- Haber listado mis actividades.

Postcondición

- Al usuario se le indica que se eliminó la actividad y desaparece de sus actividades.

Flujo Normal

1. El usuario tipo empresa se encuentra visualizando sus actividades.
2. El usuario tipo empresa pulsa sobre **X** de la actividad a eliminar.
3. Confirma la eliminación de la actividad.
4. La actividad eliminada desaparece de la lista.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible eliminar la actividad.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Modificar actividad**	**ID**	21
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario tipo empresa (del que derivan hotel y empresa).

Personal involucrado o intereses

- El usuario tipo empresa desea modificar una actividad que anteriormente añadió en el sistema.

Descripción

El usuario tipo empresa desea modificar una actividad suya del sistema para cambiar los datos que introdujo anteriormente.

Trigger

Hacer clic en **icono de pluma** sobre la actividad.

Precondición

- Haber listado mis actividades.

Postcondición

- Al usuario se le muestra una pantalla la actividad seleccionada en modo editable para que pueda modificarla y guardar los cambios.

Flujo Normal

1. El usuario tipo empresa se encuentra visualizando sus actividades.
2. El usuario tipo empresa pulsa sobre **icono de pluma** de la actividad a modificar.
3. El usuario tipo empresa modifica la actividad y guarda los cambios.
4. Al usuario tipo empresa se le notifica que se llevó correctamente a cabo el proceso.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible modificar la actividad.
2. Ir al paso 1.

1. El sistema indica que no es posible guardar los cambios de la modificación.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Añadir actividad filtrada**	**ID**	22
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario hotel

Personal involucrado o intereses

- El usuario hotel desea que a los usuarios vinculados a su hotel les aparezca la actividad seleccionada por el hotel como recomendada.

Descripción

El usuario hotel tras seleccionar una actividad de la lista de actividades, podrá añadirla a su lista de actividades filtradas. De este modo el usuario cliente vinculado al hotel verá esta actividad como recomendada.

Trigger

Pulsar sobre la actividad seleccionada en el botón **Add to suggestions**.

Precondición

- Haber listado actividades.
- Haber mostrado el detalle de la actividad a sugerir.

Postcondición

- La actividad aparece en su lista de actividades filtradas y como recomendada para los usuarios unidos al hotel.

Flujo Normal

1. El usuario hotel se encuentra listando actividades.
2. El usuario hotel visualiza el detalle de la actividad a sugerir.
3. El usuario hotel pulsa sobre **Add to suggestions.**

Flujo Alternativo

Excepción

1. El sistema reporta indica que no ha sido posible añadir la actividad filtrada.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Añadir noticia**	**ID**	23
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario hotel

Personal involucrado o intereses

- El usuario hotel desea publicar una noticia para que les llegue a los usuarios vinculados al hotel.

Descripción

El usuario hotel publicará una noticia para que los usuarios clientes la reciban en sus dispositivos o en la página web.

Trigger

Pulsar sobre la actividad seleccionada en el botón **Notification** del menú de acciones **New**.

Precondición

Postcondición

- El hotel añade la noticia a su lista de noticias y los usuarios reciben el mensaje en sus clientes.

Flujo Normal

1. El usuario hotel pulsa sobre **Notification** de su menú de acciones **New**.
2. El usuario completa los datos de la notificación.
3. Al usuario hotel se le indica que publicó correctamente la noticia.

Flujo Alternativo

Excepción

1. El sistema reporta indica que no ha sido posible añadir la noticia.
2. Ir al paso 1.

1. El sistema reporta indica que no ha sido posible añadir la noticia.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	Borrar actividad filtrada	ID	24
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario hotel

Personal involucrado o intereses

- El usuario hotel desea que una actividad que añadió anteriormente a su lista de actividades filtradas ya no aparezca.

Descripción

El usuario hotel borrará de su lista de actividades filtrada la actividad seleccionada.

Trigger

Pulsar sobre el botón **X** de la actividad a eliminar.

Precondición

- Haber añadido anteriormente la actividad como filtrada.
- Haber listado actividades filtradas.

Postcondición

- El hotel elimina de su lista de actividades filtradas la actividad seleccionada.
- Ya no le aparecerá a los usuarios unidos al hotel esa actividad como recomendada.

Flujo Normal

1. El usuario hotel pulsa en la **X** de la actividad a eliminar.
2. El usuario hotel confirma la eliminación de la actividad.
3. La actividad desaparece de la lista de actividades filtradas y se le indica al usuario que el proceso se ha llevado a cabo.

Flujo Alternativo

Excepción

1. El sistema reporta indica que no ha sido posible eliminar la actividad filtrada.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Borrar notificación**	**ID**	25
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario hotel

Personal involucrado o intereses

- El usuario hotel desea borrar una noticia que anteriormente publicó.

Descripción

El usuario hotel borrará de su lista de noticias la noticia seleccionada.

Trigger

Pulsar sobre el botón **X** de la notificación a eliminar.

Precondición

- Haber añadido anteriormente la noticia.
- Haber listado las noticias.

Postcondición

- El hotel elimina de su lista de noticias la noticia seleccionada.

Flujo Normal

1. El usuario hotel pulsa en la **X** de la actividad a eliminar.
2. El usuario hotel confirma la eliminación.
3. La noticia desaparece de la lista de actividades filtradas.

Flujo Alternativo

Excepción

1. El sistema reporta indica que no ha sido posible eliminar la noticia.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	Modificar notificación	ID	26
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario hotel

Personal involucrado o intereses

- El usuario hotel desea modificar una noticia que ya publicó.

Descripción

El usuario hotel realizará una modificación de una noticia publicada.

Trigger

Hacer clic en el **icono de la pluma** sobre la noticia que se desea modificar.

Precondición

- Haber listado las noticias.

Postcondición

- La noticia es visualizada en campos editables para poder modificarla y guardar los cambios.

Flujo Normal

1. El usuario hotel pulsa sobre el **icono de la pluma** de la actividad a modificar.
2. El usuario hotel modifica los campos editables de la noticia.
3. El usuario hotel guarda los cambios realizados en la noticia.
4. Al usuario hotel se le notifica que se modificó correctamente.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible mostrar la noticia para modificarla.
2. Ir al paso 1.

1. El sistema indica que no es posible guardar los cambios realizados en la noticia.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Ver actividades filtradas**	**ID**	27
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario hotel

Personal involucrado o intereses

- El usuario hotel desea visualizar las actividades que filtró para sus clientes anteriormente.

Descripción

El usuario hotel desea ver qué actividades filtró anteriormente y las visualiza mediante una lista.

Trigger

Hacer clic en **Suggested activities** en el panel principal.

Precondición

Postcondición

- Al usuario se le muestra una lista con las actividades que ha filtrado para su hotel anteriormente.

Flujo Normal

1. El usuario hotel visualiza las actividades que filtró anteriormente pulsando sobre **Suggested activities.**

Flujo Alternativo

Excepción

1. El sistema indica que no es posible mostrar las actividades filtradas.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	Ver notificaciones	ID	28
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario hotel.

Personal involucrado o intereses

- El usuario hotel desea visualizar las notificaciones que ha publicado anteriormente.

Descripción

El usuario hotel desea ver las notificaciones que publicó anteriormente

Trigger

Hacer clic en **Notifications** en el panel principal.

Precondición

Postcondición

- Al usuario se le muestra una lista con las notificaciones que ha publicado anteriormente.

Flujo Normal

1. El usuario hotel visualiza las notificaciones que publicó anteriormente pulsando sobre **Notifications.**

Flujo Alternativo

Excepción

1. El sistema indica que no es posible mostrar las notificaciones.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Añadir usuario**	**ID**	29
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario administrador local (sus derivados hotel y empresa administrador).

Personal involucrado o intereses

- El usuario administrador local desea añadir un nuevo usuario para su tipo empresa.

Descripción

El usuario tipo empresa desea que en su tipo empresa haya un usuario más que pueda gestionar las actividades de la tipo empresa.

Trigger

Hacer clic en **User** en el panel de acciones **New**.

Precondición

Postcondición

- Se añade un nuevo usuario al sistema que pertenece al tipo empresa por el que fue creado.

Flujo Normal

1. El usuario tipo empresa pulsa sobre **User** del menú de acciones **New.**
2. Rellena los campos de registro.
3. Guarda los datos del nuevo usuario.
4. Al usuario se le notifica que se llevó correctamente a cabo el proceso.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible añadir un nuevo usuario.
2. Ir al paso 1.

1. El sistema indica que no es posible guardar los datos del nuevo usuario
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Eliminar usuario**	**ID**	30
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario administrador local (sus derivados hotel y empresa administrador).

Personal involucrado o intereses

- El usuario administrador local desea eliminar un usuario de su tipo empresa.

Descripción

El usuario tipo empresa desea eliminar un usuario de su tipo empresa, de modo que este ese usuario queda borrado del sistema.

Trigger

Hacer clic en la **X** sobre el usuario a eliminar.

Precondición

- Haber listado los usuarios de mi tipo empresa.

Postcondición

- El usuario seleccionado queda borrado del sistema.

Flujo Normal

1. El usuario tipo empresa se encuentra visualizando los usuarios de su tipo empresa.
2. A continuación pulsa sobre el icono de la **X** del usuario a eliminar**.**
3. El usuario desaparece de la lista de usuarios de la compañía.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible eliminar el usuario seleccionado.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Modificar usuario**	**ID**	31
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario administrador local (sus derivados hotel y empresa administrador).

Personal involucrado o intereses

- El usuario administrador local desea visualizar los usuarios de su tipo empresa.

Descripción

El usuario tipo empresa desea ver los usuarios de su tipo empresa, para ello, se le muestra una lista con los usuarios registrados pertenecientes a ella.

Trigger

Hacer clic en el icono de **pincel** en el usuario editar.

Precondición

- Se muestra una lista con los usuarios del sistema pertenecientes a ese tipo empresa.

Postcondición

- El usuario queda modificado en el sistema

Flujo Normal

1. El usuario tipo empresa hace clic sobre el icono **pluma** perteneciente al usuario a editar.
2. Se modifican los datos del usuario.
3. Se pulsa sobre el botón Save.
4. Se muestra una notificación indicando que los cambios se realizaron correctamente.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible modificar los datos del usuario.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	Ver usuarios	ID	32
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario administrador local (sus derivados hotel y empresa administrador).

Personal involucrado o intereses

- El usuario administrador local desea visualizar los usuarios de su tipo empresa.

Descripción

El usuario tipo empresa desea ver los usuarios de su tipo empresa, para ello, se le muestra una lista con los usuarios registrados pertenecientes a ella.

Trigger

Hacer clic en **Users** en el panel principal.

Precondición

Postcondición

- Se muestra una lista con los usuarios del sistema pertenecientes a ese tipo empresa.

Flujo Normal

5. El usuario tipo empresa visualiza los usuarios pertenecientes a su tipo empresa pulsando en **Users** en el panel principal**.**

Flujo Alternativo

Excepción

3. El sistema indica que no es posible mostrar los usuarios.
4. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Ver estadísticas**	**ID**	33
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario administrador local (sus derivados hotel y empresa administrador).

Personal involucrado o intereses

- El usuario administrador local desea visualizar las estadísticas de su compañía.

Descripción

Uno de los usuarios de la tipo empresa pasa a ser administrador teniendo las mismas capacidades que el que le otorga ese rol.

Trigger

Hacer clic en **Statistics** sobre el panel principal.

Precondición

Postcondición

- Se muestra un una completa página con las estadísticas de la compañía.

Flujo Normal

1. El usuario tipo empresa pulsa en **Statistics.**
2. El usuario visualiza las estadísticas.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible convertir a administrador el usuario.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	Ver datos de empresa	ID	34
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario administrador local (sus derivados hotel y empresa administrador).

Personal involucrado o intereses

- El usuario administrador local desea ver los datos de la empresa a la que pertenece.

Descripción

Uno de los usuarios de la tipo empresa pasa a ser administrador teniendo las mismas capacidades que el que le otorga ese rol.

Trigger

Hacer clic en **Company profile** sobre el menú de usuario.

Precondición

Postcondición

- Se muestra el perfil de la empresa a la que se pertenece.

Flujo Normal

1. El usuario tipo empresa hace clic sobre **Company profile** del menú de usuario**.**
2. Al usuario se le muestra el perfil de la empresa.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible mostrar el perfil de la empresa.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Modificar datos de empresa**	**ID**	35
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario administrador local (sus derivados hotel y empresa administrador).

Personal involucrado o intereses

- El usuario administrador modificar los datos del perfil de la empresa

Descripción

El usuario empresa modifica el perfil de la empresa.

Trigger

Hacer clic en **Modify** en el formulario.

Precondición

- Haber mostrado el perfil de la empresa.

Postcondición

- Se modifica el perfil de la empresa con los nuevos datos.

Flujo Normal

1. El usuario modifica los datos del perfil de la empresa.
2. Guarda los cambios pulsando sobre **Modify.**
3. Se comunica que el perfil se modificó correctamente.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible modificar el perfil de la empresa.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	Borrar usuario	ID	36
Creado por	Diego Nieto Muñoz	Fecha	27/11/2012
Modif. por		Fecha Modif	

Actor principal

- Usuario administrador

Personal involucrado o intereses

- El usuario administrador desea borrar los datos de un usuario del sistema.

Descripción

El usuario administrador eliminar un usuario del sistema, para ello elimina todos sus datos.

Trigger

Hacer clic en el icono en forma de **X** sobre el usuario a eliminar.

Precondición

- Haber listado al menos un usuario.

Postcondición

- Los datos del usuario quedan eliminados del sistema.

Flujo Normal

1. Cuando visualiza los usuarios se elimina uno de ellos pulsando en el icono en forma de **X** sobre el usuario a eliminar.
2. Se confirma que se desea eliminar el usuario.
3. El usuario es eliminado del sistema.
4. El usuario desaparece de la lista.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible borrar el usuario.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Añadir usuario**	**ID**	37
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario administrador

Personal involucrado o intereses

- El usuario administrador desea añadir un usuario al sistema.

Descripción

El usuario administrador añade un usuario al sistema mediante un registro de datos.

Trigger

Hacer clic en **User** en el panel de acciones **New** de la compañía.

Precondición

Postcondición

- Se muestra crea un nuevo usuario en el sistema.

Flujo Normal

1. Se hace clic sobre **User** del panel de acciones **New**.
2. Se rellenan los datos de registro del usuario.
3. Se guardan los datos de para finalizar el registro del nuevo usuario.
4. Se recibe una notificación que indica que todo fue correctamente.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible añadir el usuario.
2. Ir al paso 1.

1. El sistema indica que no es posible guardar los cambios de añadir el usuario.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Modificar usuario**	**ID**	38
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario administrador

Personal involucrado o intereses

- El usuario administrador desea modificar los datos de un usuario del sistema.

Descripción

El usuario administrador modifica los datos de un usuario del sistema mediante la visualización y cambio de los datos registrados en el sistema.

Trigger

Hacer clic en el icono de la **pluma** sobre el usuario a editar.

Precondición

- Haber listado los usuarios.

Postcondición

- Los datos del usuario quedan modificados y guardados en el sistema.

Flujo Normal

1. Cuando visualiza los usuarios se modifica uno de ellos pulsando el icono **pluma** sobre el usuario a editar.
2. Se modifican los datos de registro del usuario.
3. Se guardan los datos de registro para finalizar la modificación del usuario.
4. Al usuario se le notifica que se llevó correctamente a cabo el proceso.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible modificar el usuario.
2. Ir al paso 1.

1. El sistema indica que no es posible guardar los cambios de modificar el usuario.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Buscar usuario**	**ID**	39
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario administrador

Personal involucrado o intereses

- El usuario administrador desea buscar un usuario en el sistema.

Descripción

El usuario administrador visualizar los usuarios del sistema que coinciden con el criterio de búsqueda introducido.

Trigger

Hacer clic en **Search** en el panel de búsqueda de usuarios.

Precondición

Postcondición

- Se listan los usuarios que coincidan con los parámetros de búsqueda.

Flujo Normal

1. Introducir el nombre del usuario a buscar.
2. Se buscan usuarios que coincidan con los parámetros de búsqueda pulsando en **Search.**
3. Se listan los usuarios que coinciden con los parámetros de búsqueda.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible buscar usuarios.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Iniciar sesión**	**ID**	40
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario anónimo.

Personal involucrado o intereses

- El usuario anónimo inicia sesión para ejercer otro rol de usuario registrado dentro del sistema.

Descripción

El usuario anónimo inicia sesión y pasa a ser algún tipo de usuario registrado con las funciones que ello tenga.

Trigger

Hacer clic en **Sing in** en la barra superior.

Precondición

Postcondición

- El usuario anónimo adquiere un nuevo rol en el sistema de tipo usuario registrado.

Flujo Normal

1. El usuario anónimo pulsa sobre el botón **Sign in** de la barra superior.
2. El usuario introduce sus credenciales en el formulario.
3. El usuario se identifica en el sistema pulsando en **Sign in.**
4. El usuario cambia el rol en el sistema a alguno de tipo usuarios registrado.

Flujo Alternativo

Excepción

1. El sistema indica que no es posible iniciar sesión.
2. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Nombre	**Registrarse**	**ID**	41
Creado por	Diego Nieto Muñoz	**Fecha**	27/11/2012
Modif. por		**Fecha Modif**	

Actor principal

- Usuario anónimo.

Personal involucrado o intereses

- El usuario anónimo desea registrarse en el sistema para poder desempañar las funciones de alguno de los roles de usuario registrado.

Descripción

El usuario anónimo inicia sesión y pasa a ser algún tipo de usuario registrado con las funciones que ello tenga.

Trigger

Hacer clic en **Sign up, Register my hotel o Register my company** en el panel principal.

Precondición

- El identificador de registro en el sistema que va a usar el usuario no puede estar en uso (mail, nickname, etc).

Postcondición

- Los datos de registro quedan guardados en el sistema y el usuario puede acceder a él con el nuevo rol creado.

Flujo Normal

1. El usuario pulsa sobre el botón **Sign up o Register my hotel o Register my company**.
2. El usuario se registra en el sistema introduciendo los datos básicos de registro en el sistema pulsando en **Register.**
3. El sistema valida los datos que ha introducido para registrarse. Si falla vuelve a 1.
4. Al usuario se le notifica que se terminó correctamente el proceso.

Flujo Alternativo

1. En caso de registrar una empresa o una compañía deben introducirse también los datos de la misma.

Excepción

2. El sistema indica que no es posible registrarse.
3. Ir al paso 1.

Includes

Requisitos Especiales

Notas

Anexo B. Manual de usuario del portal web.

La aplicación web desarrollada posee una gran cantidad de funciones y opciones cuyo uso será detallado a continuación por roles.

Rol de invitado

Interfaz principal
Cuando se accede a EIAT nos encontramos con un portal web que describe el sistema y sus funcionalidades.

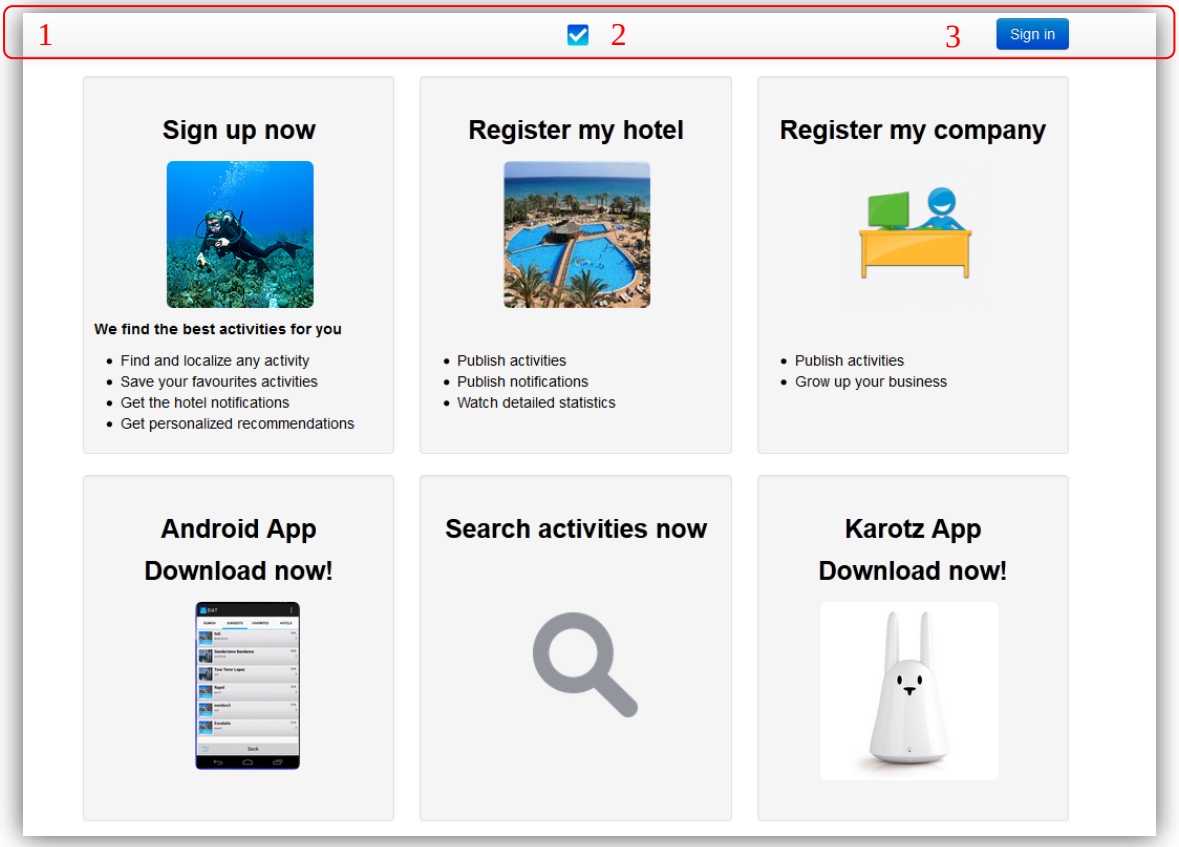


Imagen 13.1. Interfaz de rol invitado

En la imagen superior podemos observar como la interfaz posee una barra de navegación fija[1], en esta barra aparecerán todos los elementos necesarios para que los usuarios de todos los roles puedan interactuar de forma sencilla con el sistema. Los botones [2] y Sign in[3] permite ir al panel principal e iniciar sesión en el sistema respectivamente.

A continuación, la pantalla se divide en seis módulos, en la parte superior se puede acceder a las funciones de registro de clientes, hoteles y empresas. En la parte inferior nos encontramos con enlaces a la aplicación cliente de Android, el buscador de actividades y la aplicación cliente de Karotz.

Registro de un cliente

El registro en la aplicación web sólo requiere unos cuantos campos. El nombre completo, nuestro correo y una contraseña. También es necesario introducir un código de verificación de interfaz humana. Una vez rellenados todos los campos sólo es necesario pulsar en *Register*.

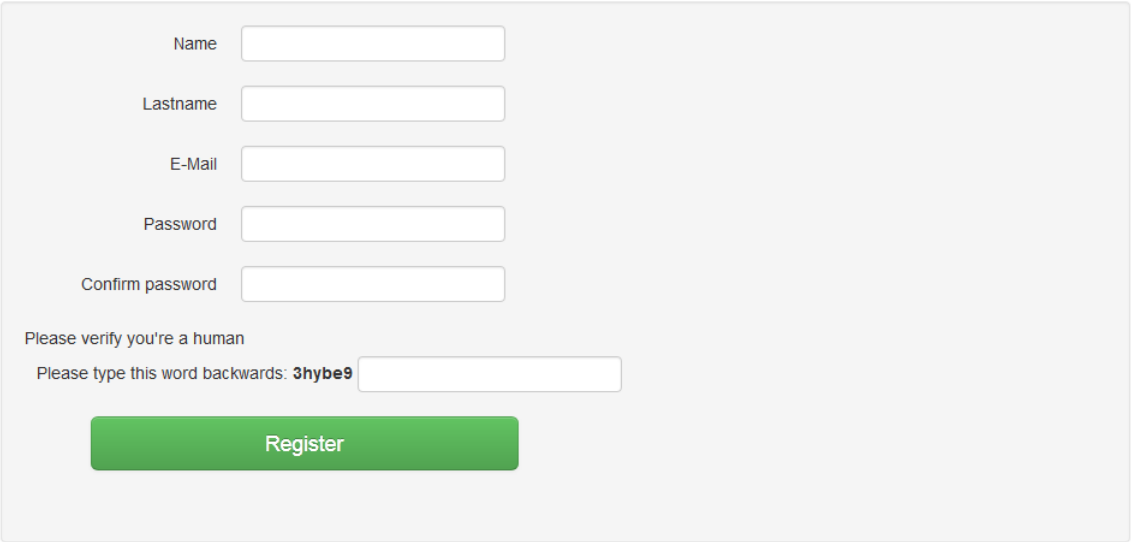


Imagen 13.2. Registro de un usuario cliente

Si todo va bien recibiremos el siguiente mensaje y ya podremos acceder a la aplicación para hacer uso de todas sus funciones.

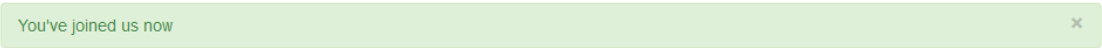


Imagen 13.3. Mensaje de registro correcto

En caso de nos hayamos equivocado en algún campo o estuvieran incompletos nos aparecerá un mensaje indicándonoslo y debajo de los campos erróneos se nos ofrecerá una explicación del error.

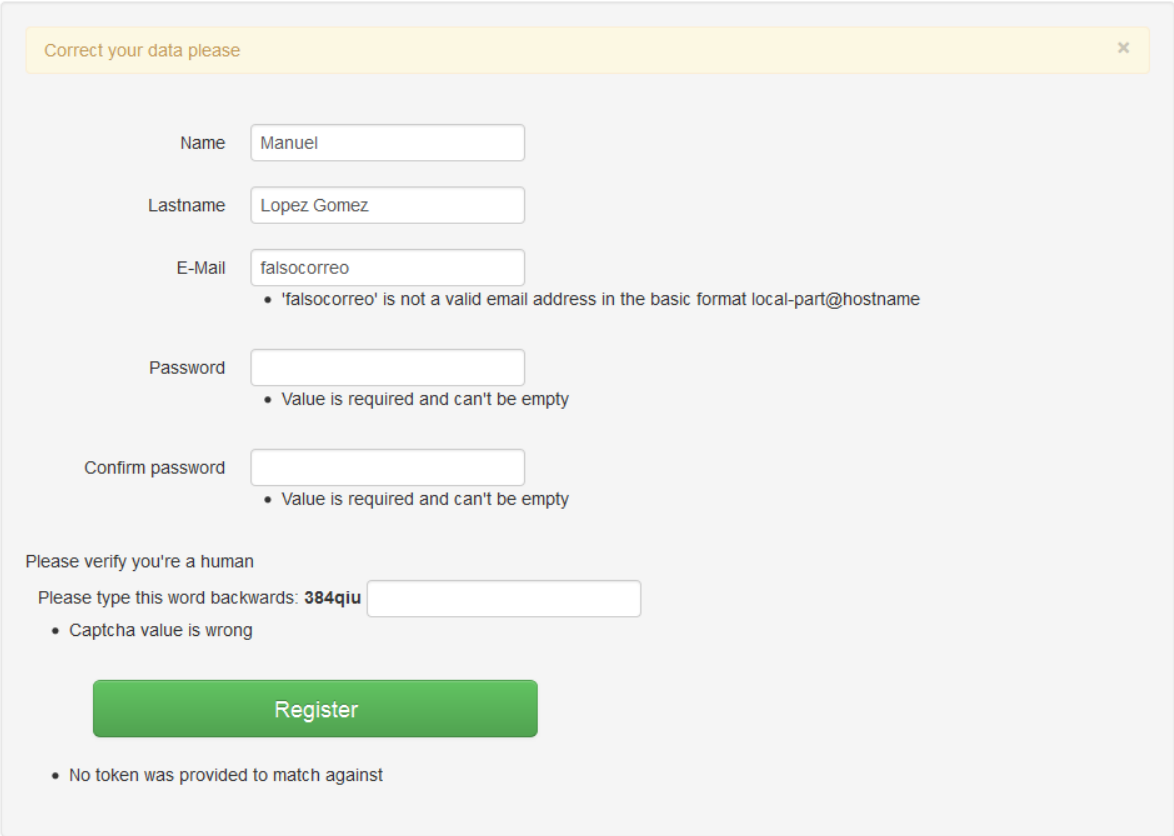


Imagen 13.4. Mensaje de fallo en el registro y formulario

Registro de una empresa u hotel

Para registrar una empresa u hotel debemos hacer clic sobre el botón correspondiente en el panel principal. A continuación deberemos rellenar unos campos correspondientes a la información de la compañía y pulsar sobre *Register*. Finalmente registramos al primer usuario administrador.

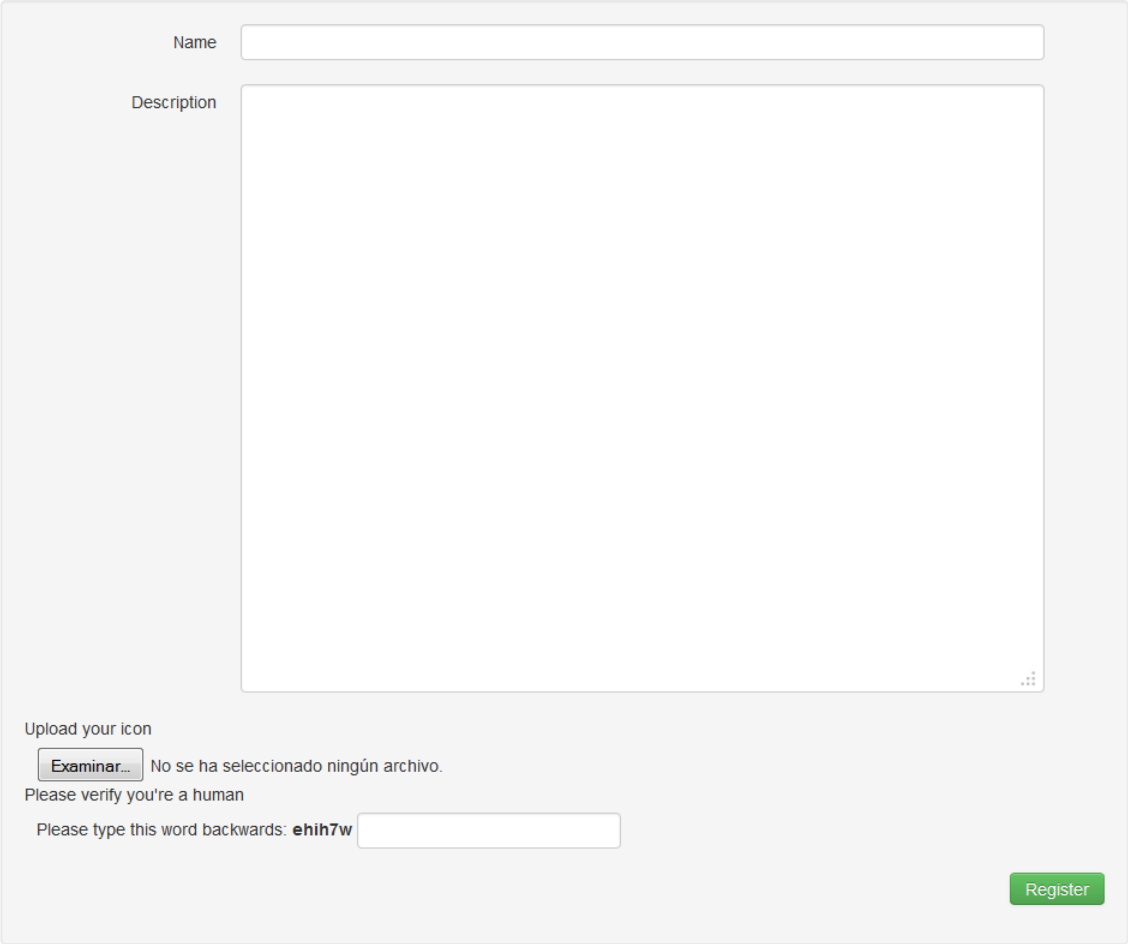


Imagen 13.5. Registro de una compañía

Acceso

Para realizar el inicio de sesión en la aplicación debemos pulsar sobre el botón *Sign in* localizado anteriormente y nos aparecerá el cuadro siguiente

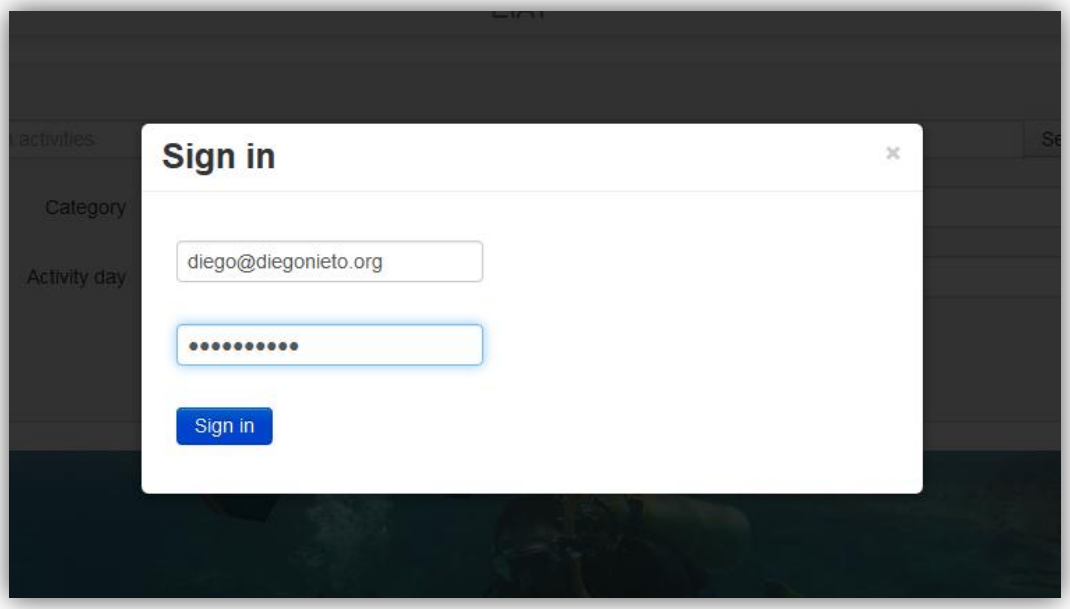


Imagen 13.6. Formulario de acceso

En este cuadro debemos ingresar el correo electrónico y la contraseña con la cual nos registramos anteriormente en la aplicación. A continuación pulsando sobre *Sign in* quedaremos logueados en el sistema si los datos son correctos.

En el caso de los datos introducidos sean incorrectos nos aparecerá la siguiente pantalla para corregirlos y acceder

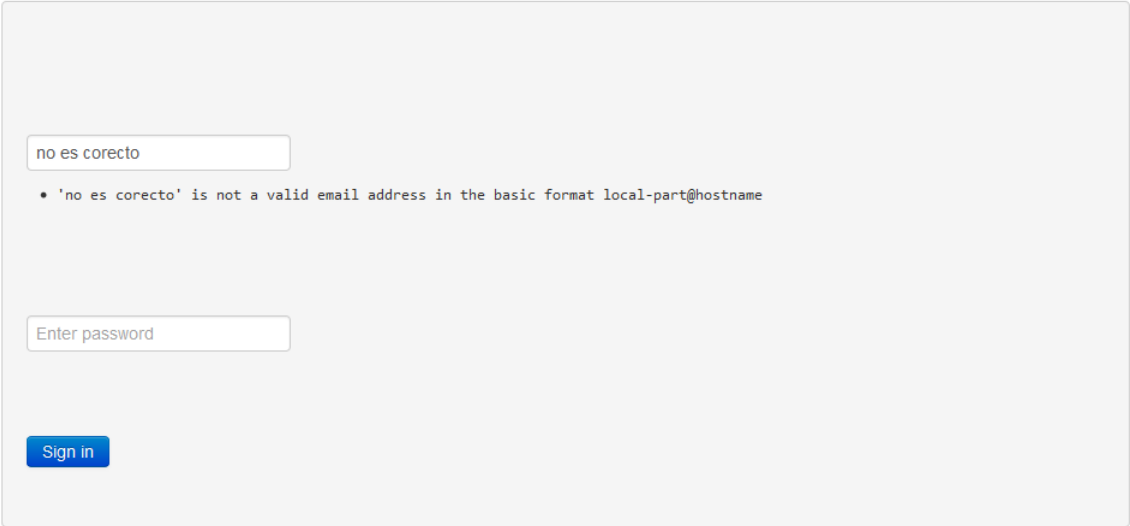


Imagen 13.7. Formulario de acceso fallido

Detalle de una actividad

El detalle de actividad de un usuario anónimo no incluye ninguna opción. En este caso tras pulsar para ver el detalle de la actividad podremos ver la compañía a la que pertenece, desde cuando hasta cuando está disponible, qué horario tiene, el precio, el teléfono de contacto, su precio, los días semanales que está disponible y su localización geográfica.

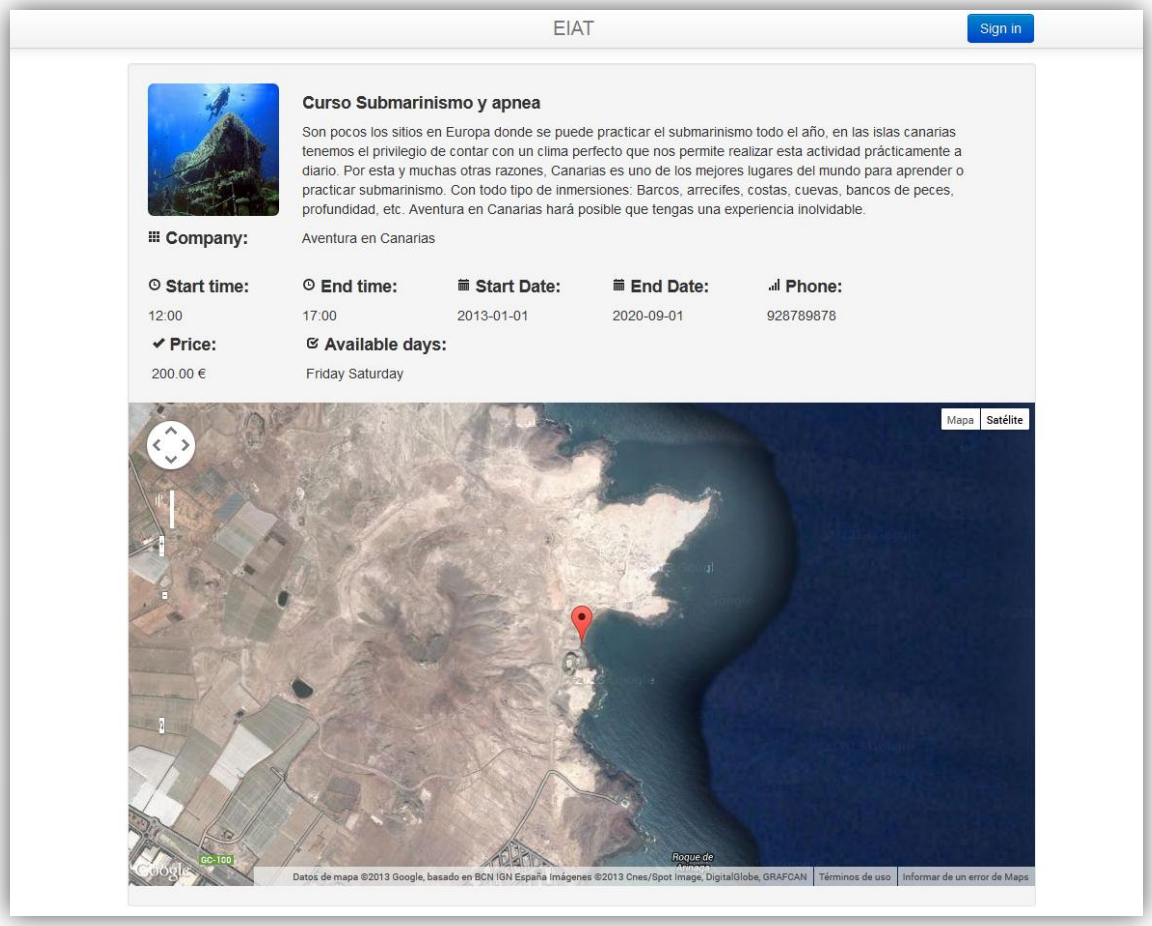


Imagen 13.8. Detalle de actividad para el rol anónimo

Rol de usuario

De acuerdo a la jerarquía y estructura de roles del sistema, este apartado contiene las dos funciones comunes a los usuarios de cualquier rol del sistema.

Listados de actividades

Los listados de actividades para los roles anónimo y usuarios registrados son iguales, se muestra una página con una determinada lista de actividades que incluye su nombre, foto y descripción.

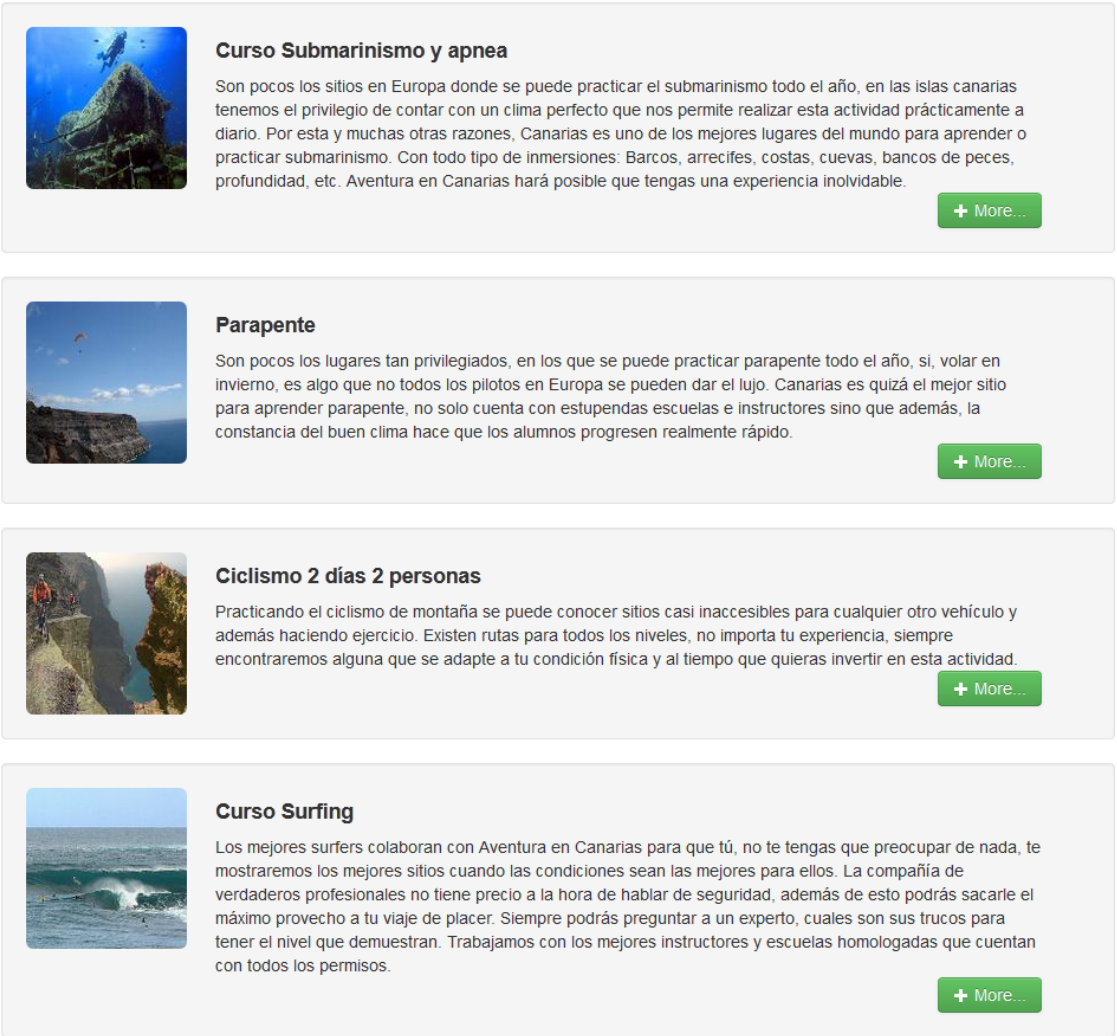


Imagen 13.9. Listado de actividades

Ver actividades populares

Haciendo clic en *Top 10 activities*[1] es posible visualizar las diez actividades más populares para los usuarios. La pantalla que se mostrará será una lista con esas actividades con las que el usuario podrá interactuar para verlas en detalle.

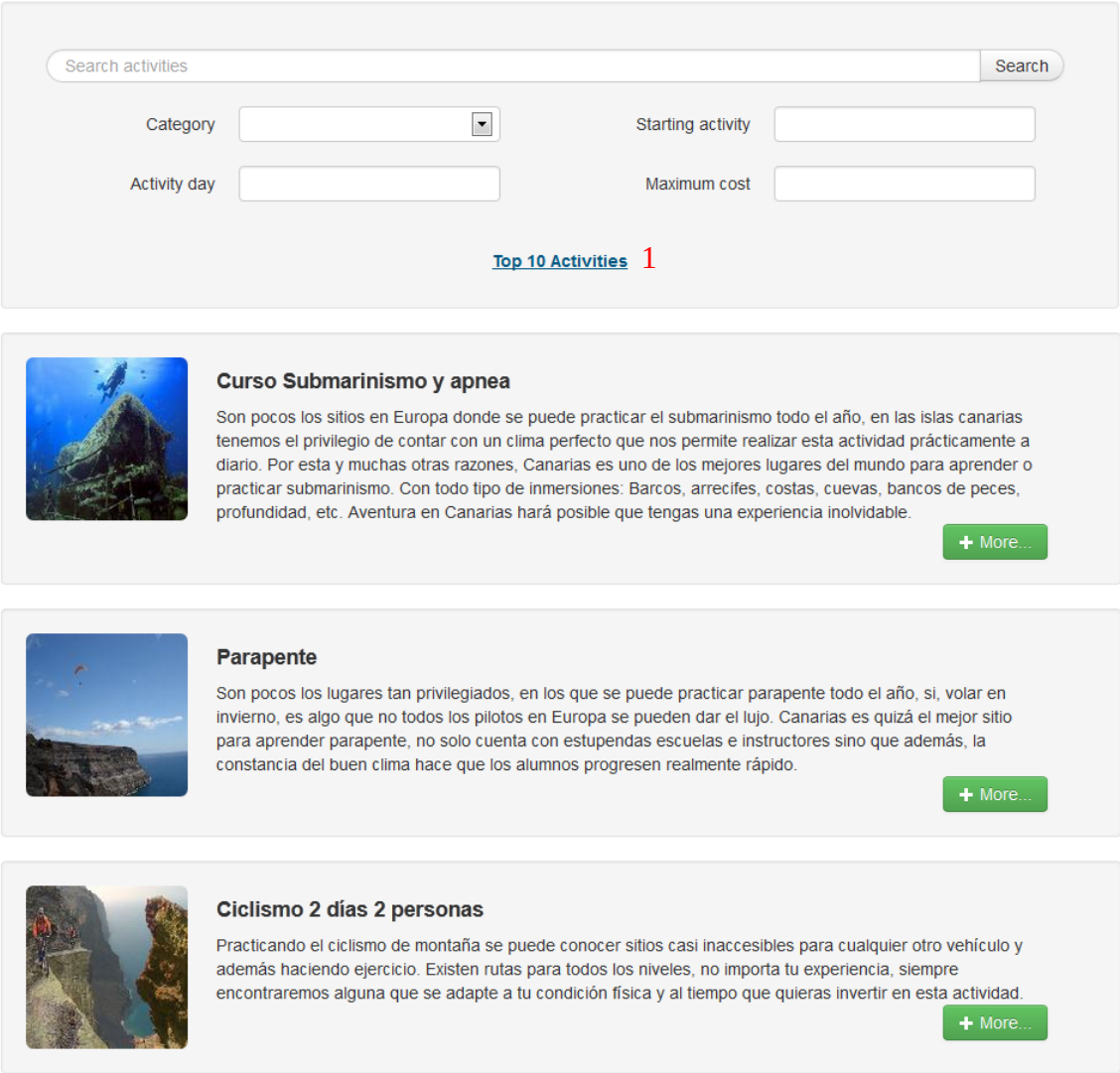


Imagen 13.10. Interfaz de *Top 10 activities*

Búsqueda de actividades

El módulo de búsqueda de actividades es uno de los más importantes de la aplicación y por ello se mantiene visible durante en casi todo momento. La búsqueda es parametrizable y es posible filtrar resultados utilizando el nombre de la actividad, la categoría a la que pertenece, el día en que se quiere realizar la actividad, la hora a la que se desea comenzarla y el precio. Todos los parámetros

de la búsqueda son opcionales, aunque por supuesto es requerido introducir al menos uno de ellos.

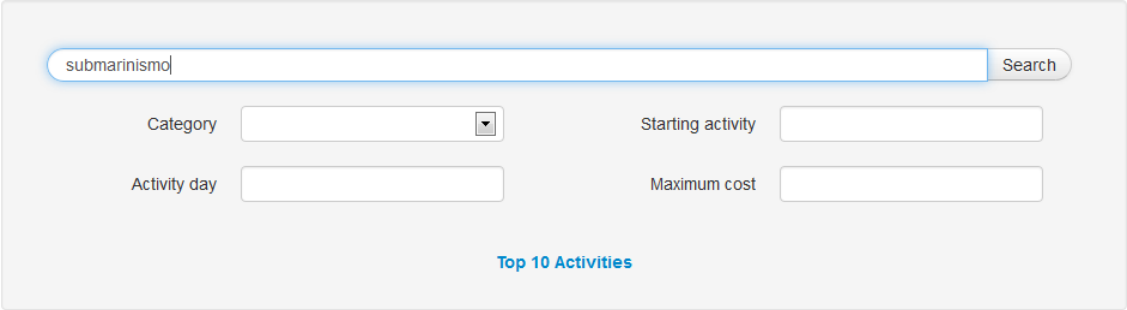


Imagen 13.11. Módulo de búsqueda. Campo de texto

Las categorías se muestran como un listado del cuadro de búsqueda y permiten que el usuario afine más el tipo de actividad a buscar.

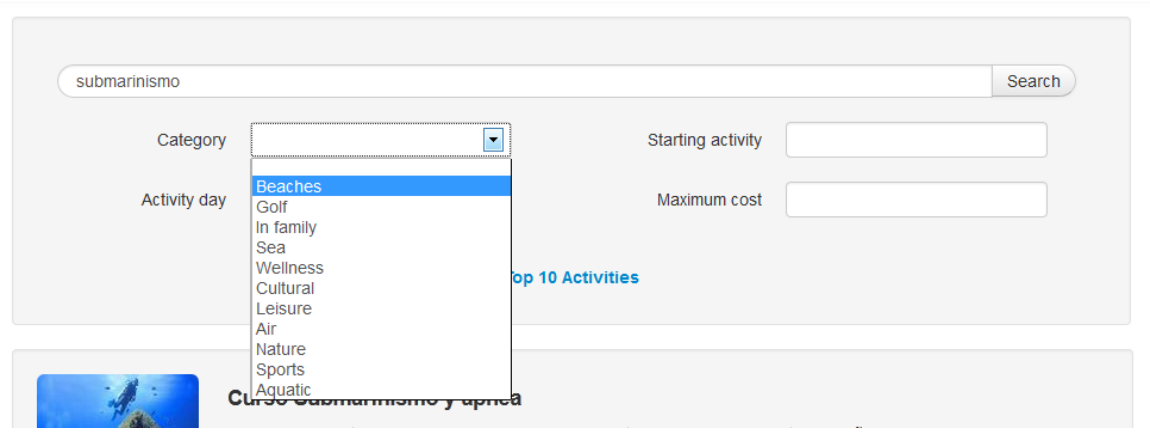


Imagen 13.12. Módulo de búsqueda. Categoría

Para que al usuario le resulte más cómo elegir una fecha y una hora se han incorporado los widgets *datepicker* y *timepicker* que permiten rellenar estos campos gráficamente.

En el *datepicker* podemos desplazarnos por los meses haciendo clic en las flechas en las esquinas superiores y elegir el día seleccionándolo.

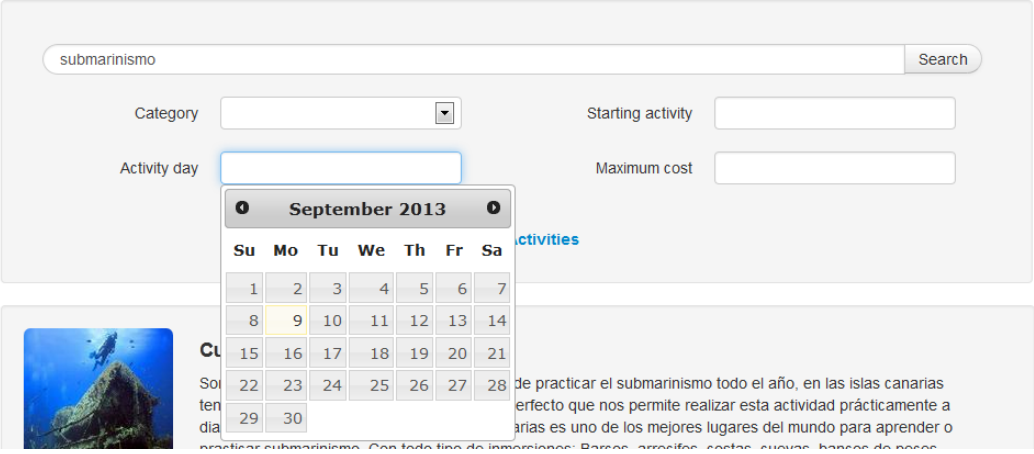


Imagen 13.13. Módulo de búsqueda. Datepicker

En el *timepicker* debemos desplazar los selectores por las barras para elegir una hora o pulsar sobre *now* para seleccionar la hora actual.

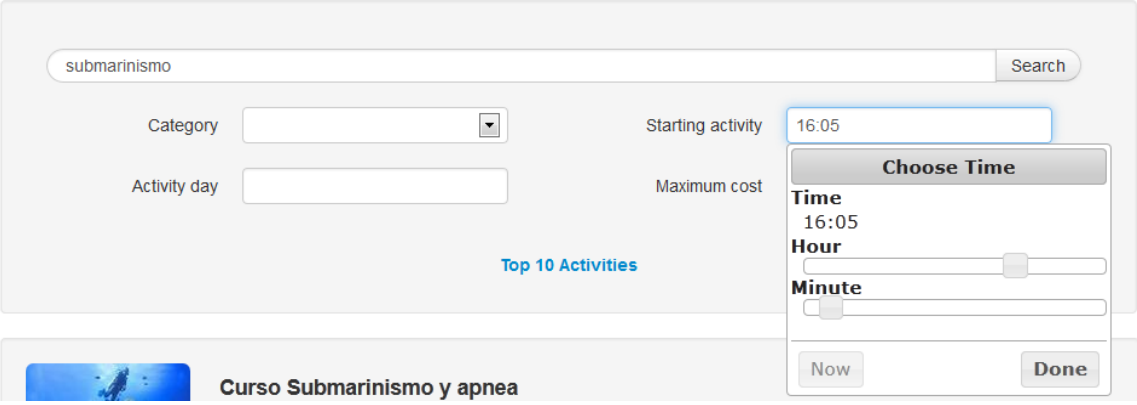


Imagen 13.14. Módulo de búsqueda. Timepicker

Finalmente, el último parámetro de búsqueda, el precio, muestra un *tooltip* que nos ayudará a introducir el precio en el formato correcto.

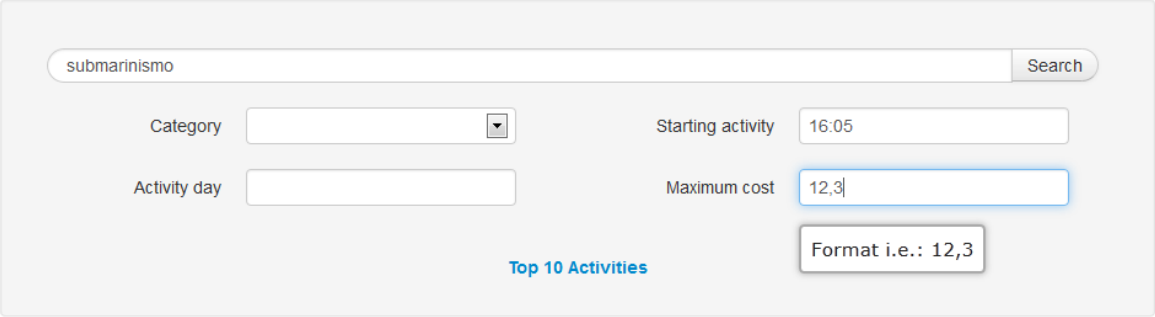


Imagen 13.15. Módulo de búsqueda. Tooltip precio

Para realizar la búsqueda debemos pulsar sobre *Search*. Los resultados serán mostrados en forma de lista, que será paginada cada cinco actividades.

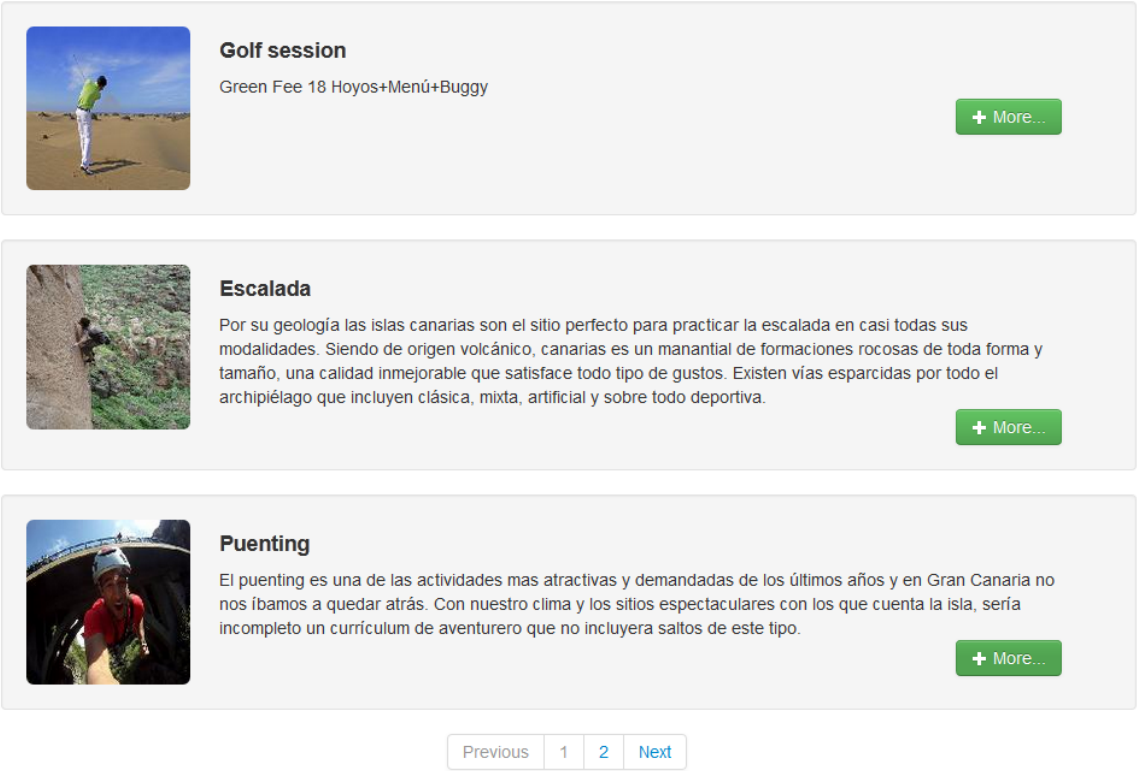


Imagen 13.16. Módulo de búsqueda. Resultados y paginación

Spinner

El Spinner incrustado en la barra superior permite indicar al usuario que se están cargando datos del servidor y que en breve aparecerán. Este elemento se muestra cuando se realizan acciones como mostrar el detalle de una actividad y sustituye al logotipo EIAT mientras se cargan los datos. Cuando la carga finaliza desaparece y se restablece el logotipo.

Imagen 13.17. Spinner

Ver el detalle de una actividad

Para visualizar el detalle de una actividad debemos hacer clic sobre el botón *More…* de la actividad en cuestión y se nos mostrará un detalle de actividad con unas opciones de acuerdo al rol que tenemos.

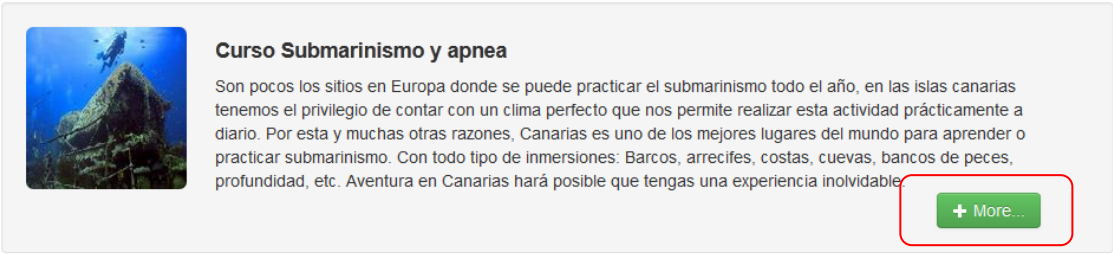


Imagen 13.18. Lista actividades. Mostrar más

Rol de usuario registrado

Todos los usuarios registrados en el sistema tienen dos funciones en común:

- Cerrar sesión
- Ver y modificar sus datos de usuario

Ver y modificar los datos de usuario

Para visualizar y modificar los datos del usuario que ha iniciado sesión se debe hacer clic sobre el botón *Profile* del menú de usuario situado en la barra superior de navegación

Tras pulsar este botón se nos mostrará un formulario con nuestros datos que podremos modificar. Además, incluye una opción para modificar la contraseña actual.

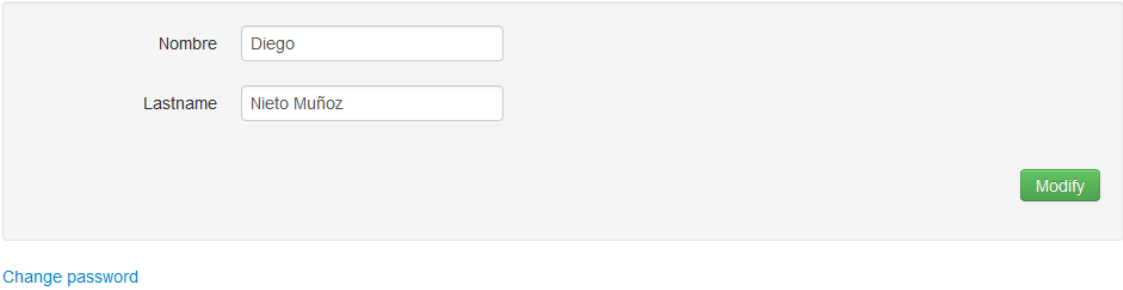


Imagen 13.19. Ver o modificar el perfil del usuario

Si pulsamos sobre Change password se mostrará el siguiente formulario donde debemos indicar la contraseña actual y la nueva:

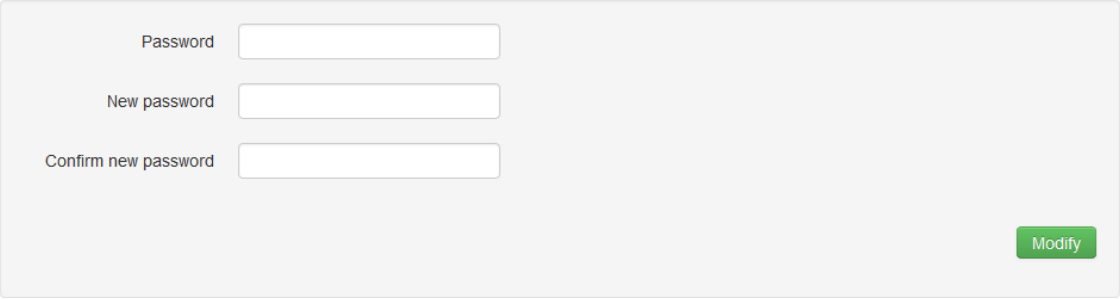


Imagen 13.20. Modificar contraseña del usuario

Rol de cliente

Panel principal y recomendación

En el panel principal del usuario cliente se muestra la recomendación que el sistema ha calculado que es la mejor para él. Por lo tanto para recibir la recomendación del sistema sólo es necesario hacer clic sobre el logo EIAT para ir al panel principal. Este panel muestra el siguiente aspecto junto con el módulo de búsqueda que en esta imagen queda oculto.

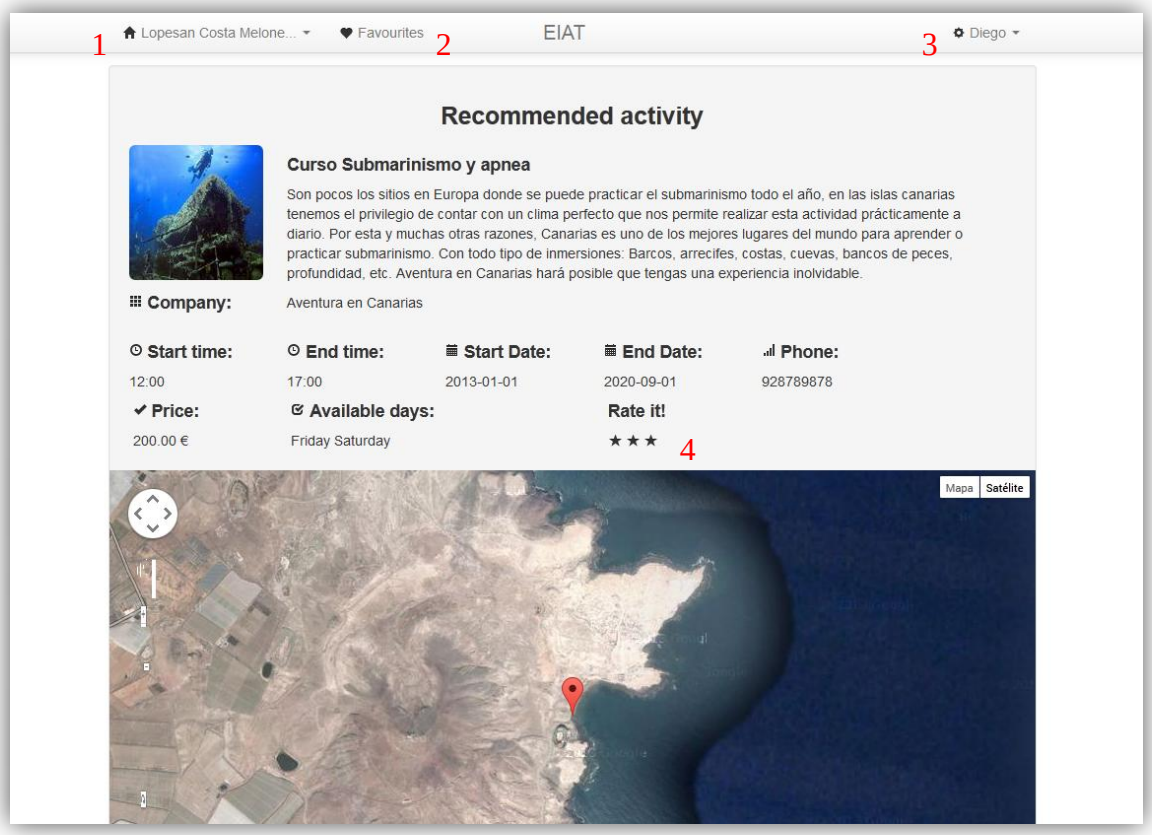


Imagen 13.21. Panel principal del usuario cliente

En la barra superior nos encontramos con varios botones que componen la funcionalidad para el cliente.

El que se encuentra más a la izquierda[1] permite desplegar el menú de acciones del hotel al que se está unido. Cuando estemos unidos a un hotel el nombre del botón será el nombre del hotel y cuando no, el nombre será *Hotels*.

El botón *Favourites*[2] nos lleva a la página que nos muestra una lista de actividades que previamente hemos marcado como favoritas.

A la derecha de la barra superior nos encontramos con el botón de acciones del usuario[3], cuyo nombre es el del usuario. Tras pulsar en él se nos muestra el menú de usuario.

También es posible votar la actividad[4] recomendada haciendo clic sobre el número de estrellas que deseemos otorgarle.

Detalle de una actividad

Los detalles de la actividad seleccionada para un usuario cliente tendrán los mismos elementos que el de usuario anónimo pero incluye otras funcionalidades como agregar a favoritos, borrar de favoritos y votar la actividad.

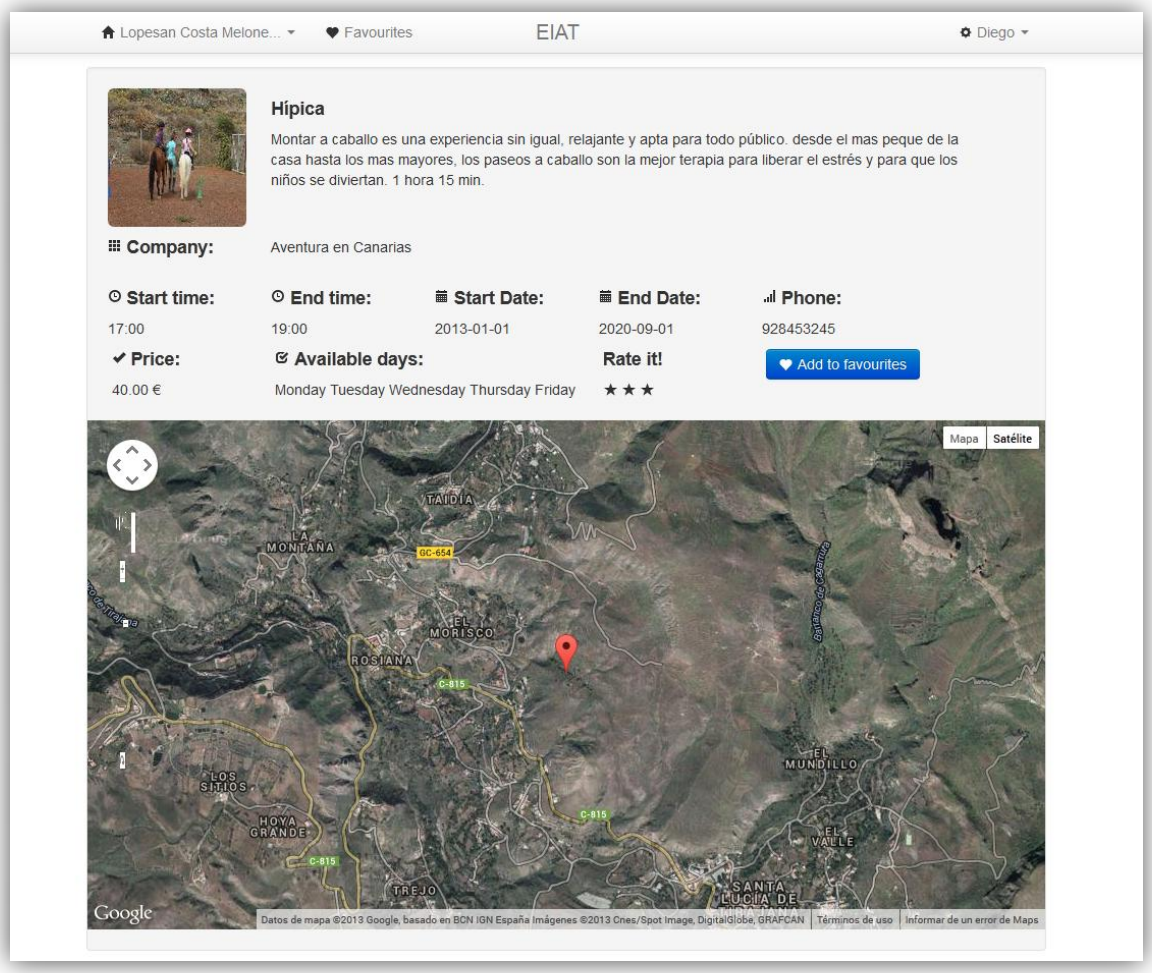


Imagen 13.22. Detalle de una actividad de un usuario cliente

Agregar y borrar de favoritos

Cuando un usuario cliente despliega el detalle de una actividad tendrá las opciones de agregar o borrar de favoritos debajo de *Phone*. La opción de agregar a favoritos aparecerá si esta actividad no está ya agregada y viceversa.

Con tan sólo hacer clic en *Add to favourites* la actividad quedará guardada.

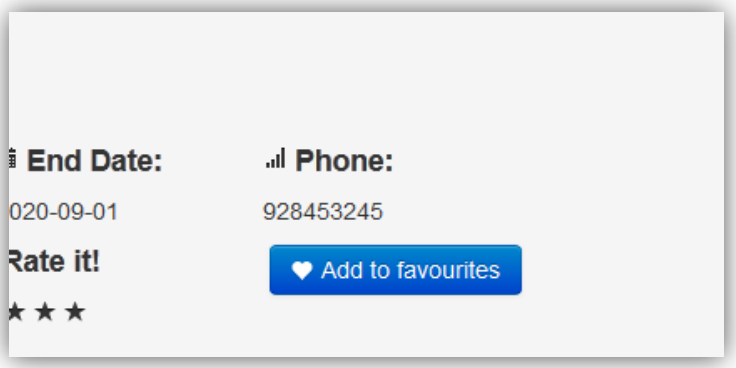


Imagen 13.23. Añadir actividad a favoritos

Para eliminar la actividad de favoritos ocurre lo mismo que para agregarla, con sólo hacer clic en *Unfavourite* la actividad quedará fuera de la lista de favoritos del usuario.

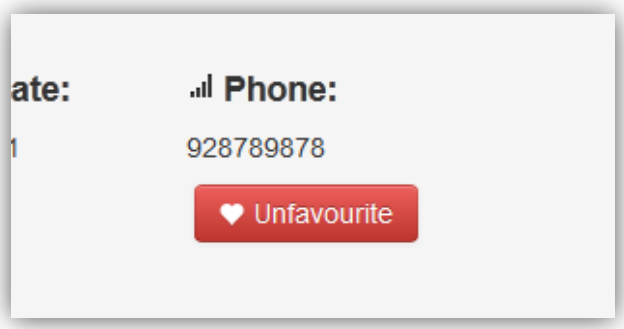


Imagen 13.24. Eliminar actividad de favoritos

Votar la actividad

El rating de una actividad es otra funcionalidad de la que dispone el usuario cliente. La funcionalidad de rating esta incrustada en el detalle de cada actividad. Para votar una actividad sólo es necesario hacer clic son la primera, segunda o tercera estrella y así darle uno, dos o tres puntos respectivamente.

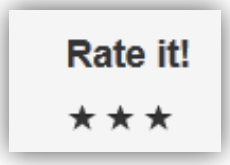


Imagen 13.25. Votar una actividad

Cuanto mayor sea el número de estrellas mayor es nuestra valoración por la actividad.

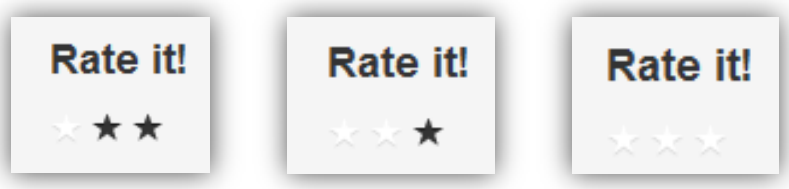


Imagen 13.26. Votar una actividad. Número de estrellas

Acciones del hotel al que se está unido

Cuando el usuario está unido a un hotel en el lugar del botón *Hotels* aparece otro botón desplegable con las acciones del hotel. Estas acciones son:

- Ver notificaciones del hotel
- Ver actividades del hotel
- Ver actividades sugeridas por el hotel
- Desunirse del hotel

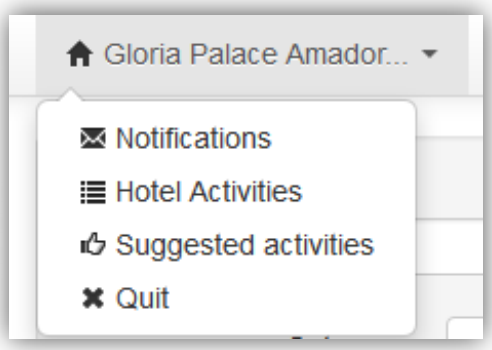


Imagen 13.27. Acciones del hotel del cliente sin notificaciones

Cuando el hotel publica alguna notificación, el número de notificaciones no leídas aparecerá al lado de *Notificacions* de la siguiente manera:

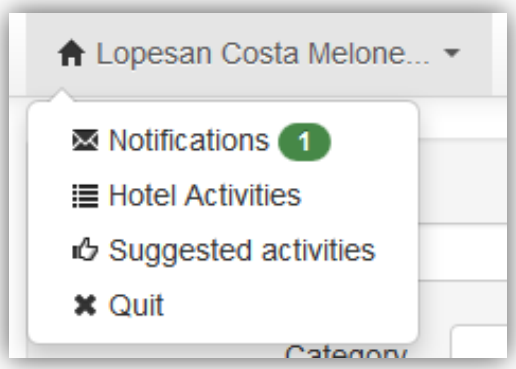


Imagen 13.28. Acciones del hotel del cliente con notificaciones

Listado de notificaciones

Si hacemos clic sobre *Notifications* del menú de acciones del hotel al que se está unido iremos a una página donde se nos mostrarán las notificaciones en orden cronológico descendente.

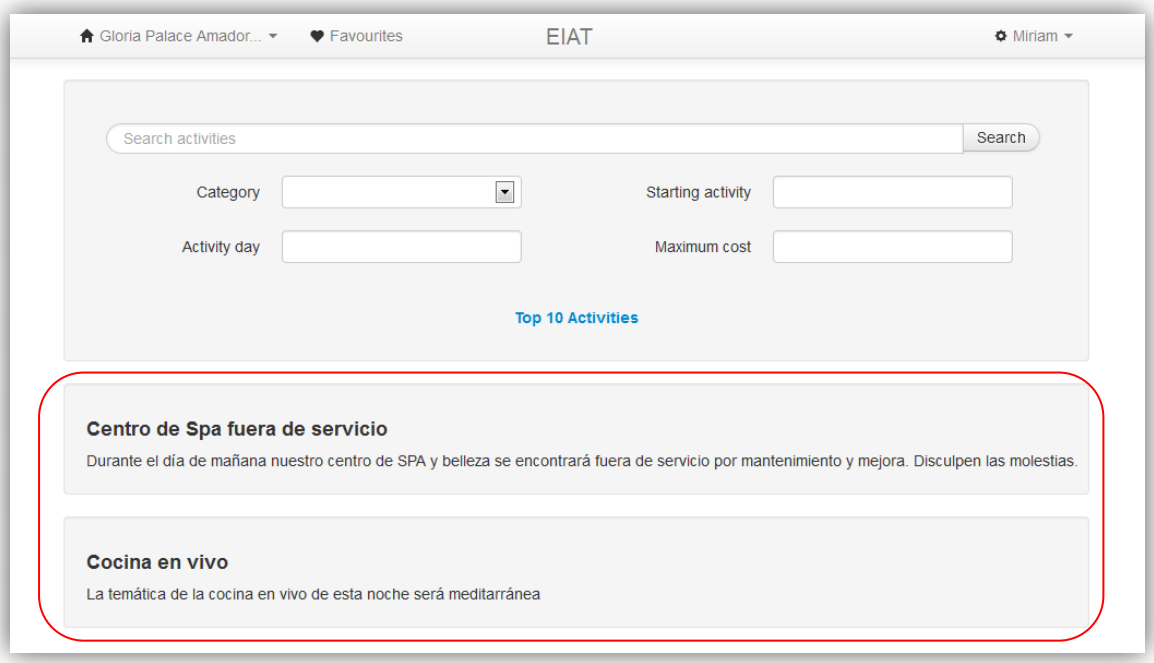


Imagen 13.29. Listado de notificaciones

Búsqueda y unión a hoteles

Cuando un usuario no está unido a un hotel en la parte izquierda de barra superior de navegación aparecerá el botón *Hotels*.

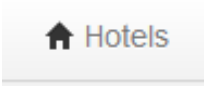


Imagen 13.30. Botón *Hotels* de la barra superior de navegación

Si hacemos clic sobre él iremos a una página donde se nos mostrará un cuadro de búsqueda[1] y varios hoteles[2]. Si en esa lista se encuentra nuestro hotel haremos clic directamente en *Join*[3] y si no buscaremos el hotel mediante el buscador. Cuando aparezca el hotel buscado simplemente debemos hacer clic en el botón *Join* del hotel correspondiente.

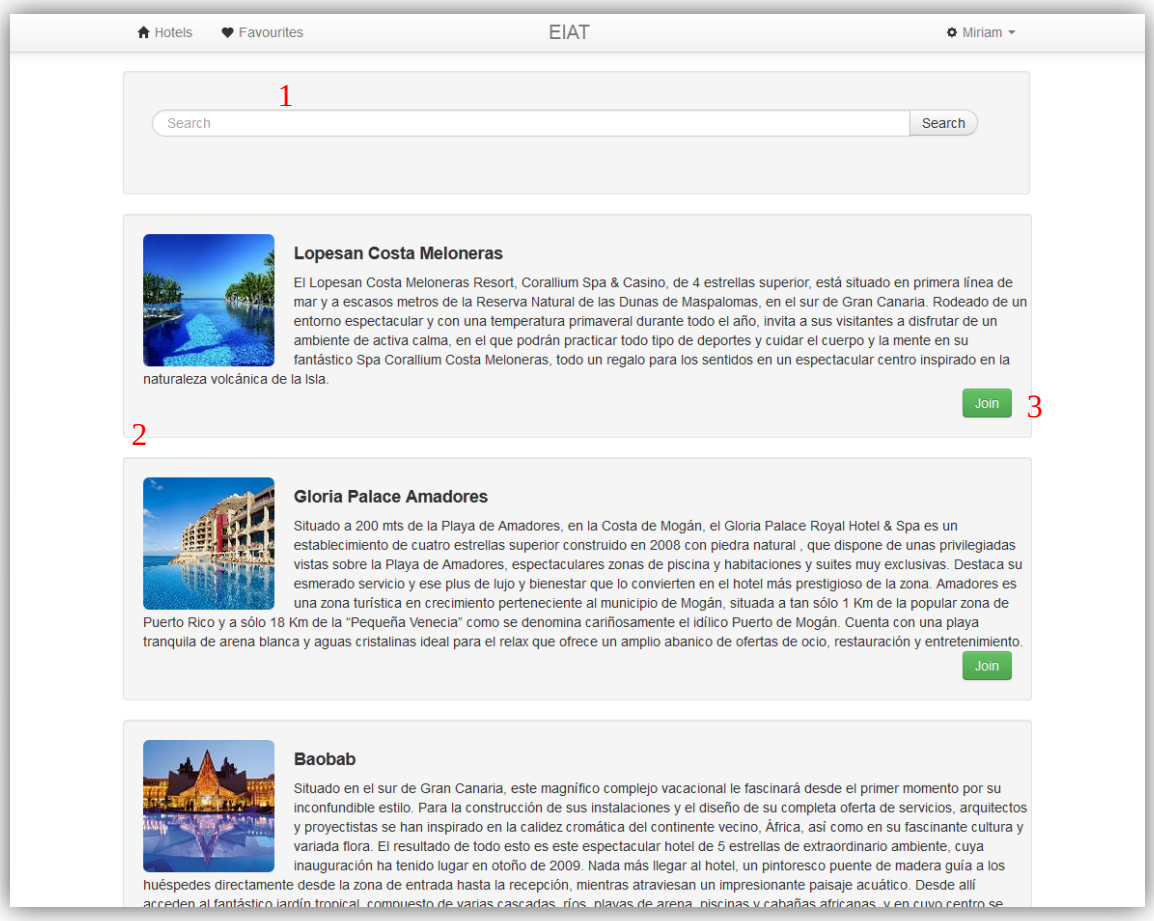


Imagen 13.31. Búsqueda y unión a hoteles

Menú de usuario

El menú de usuario está posicionado a la derecha de la barra superior de navegación y permite acceder a las siguientes funciones:

- Ver y modificar las preferencias.
- Ver y modificar los datos del usuario.
- Cerrar la sesión.

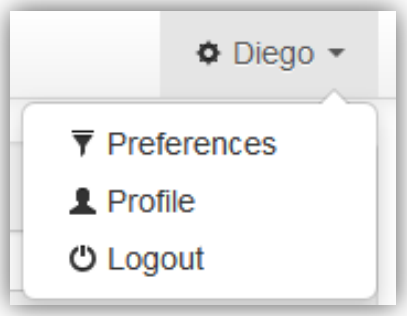


Imagen 13.32. Menú de usuario cliente

Visualizar y modificar preferencias

Cuando deseemos ver o modificar nuestras preferencias sólo es necesario hacer clic sobre *Preferences* del menú de usuario y se nos mostrará un cuadro como el siguiente:

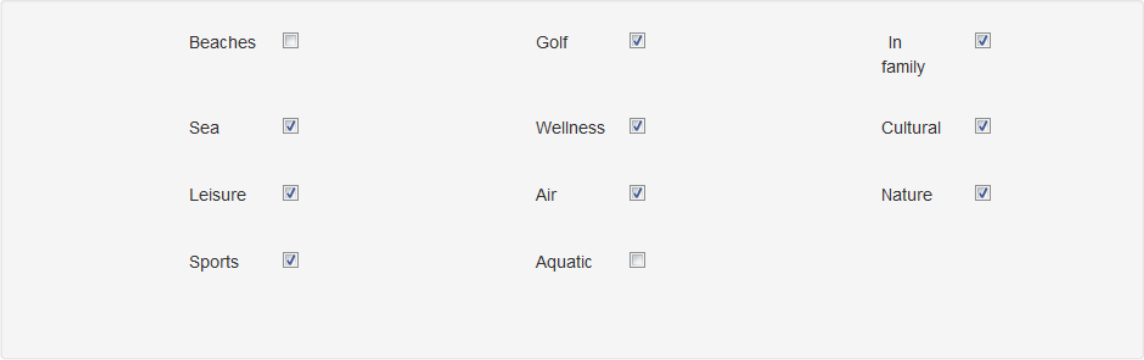


Imagen 13.33. Ver y modificar preferencias

Rol de hotel

Panel principal

Cuando un usuario con el rol de hotel inicia sesión en la aplicación accede a su panel principal. Desde él puede crear, ver, modificar y borrar las actividades y notificaciones del hotel. También puede borrar las actividades que está sugiriendo a sus huéspedes.

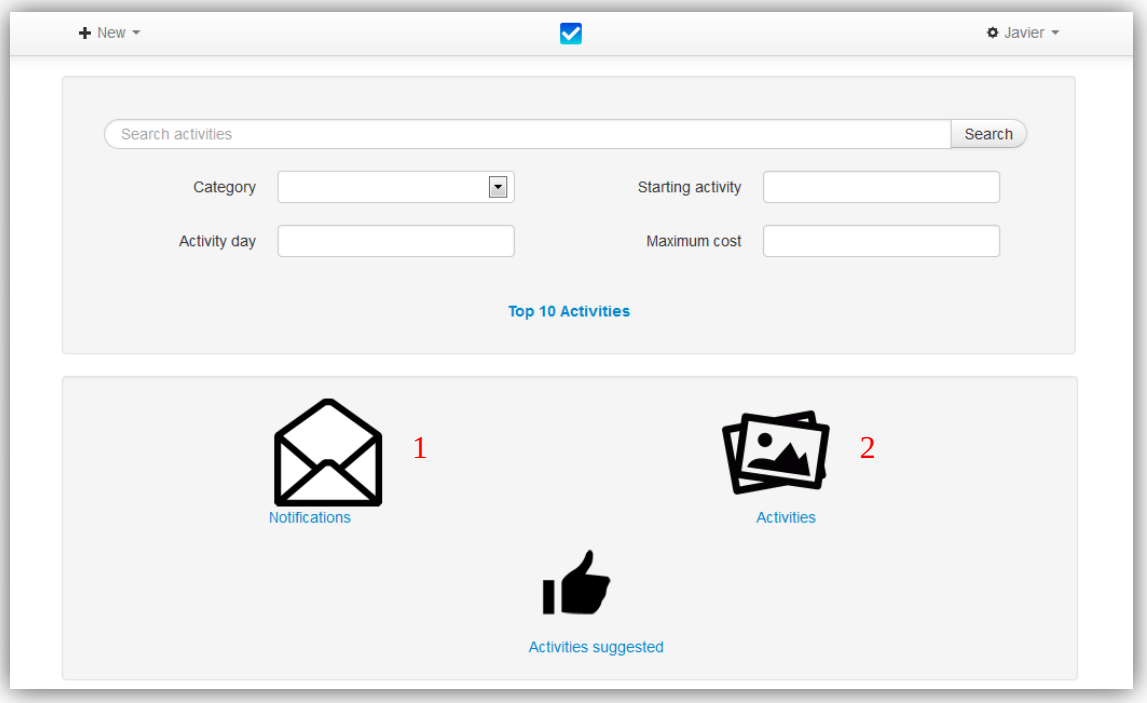


Imagen 13.34. Panel principal del usuario hotel

Para ver, modificar y borrar sus notificaciones y actividades debe hacer clic sobre el icono o nombre de notificaciones[1] o actividades[2], según corresponda. Para crear nuevas actividades o notificaciones debe acceder al menú de acciones del hotel *New*[3] y pulsar sobre *Notification*[4] o *Activity*[5].

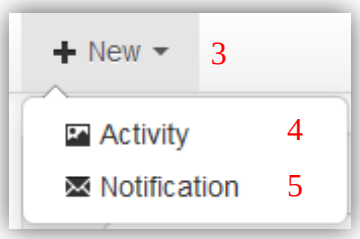


Imagen 13.35. Menú de acciones del usuario hotel

Nueva notificación

Para crear una nueva notificación el usuario hotel u hotel administrador debe hacer clic sobre *New* del menú de acciones del hotel y después sobre *Notification*. A continuación se mostrará el siguiente formulario para crear una nueva notificación.

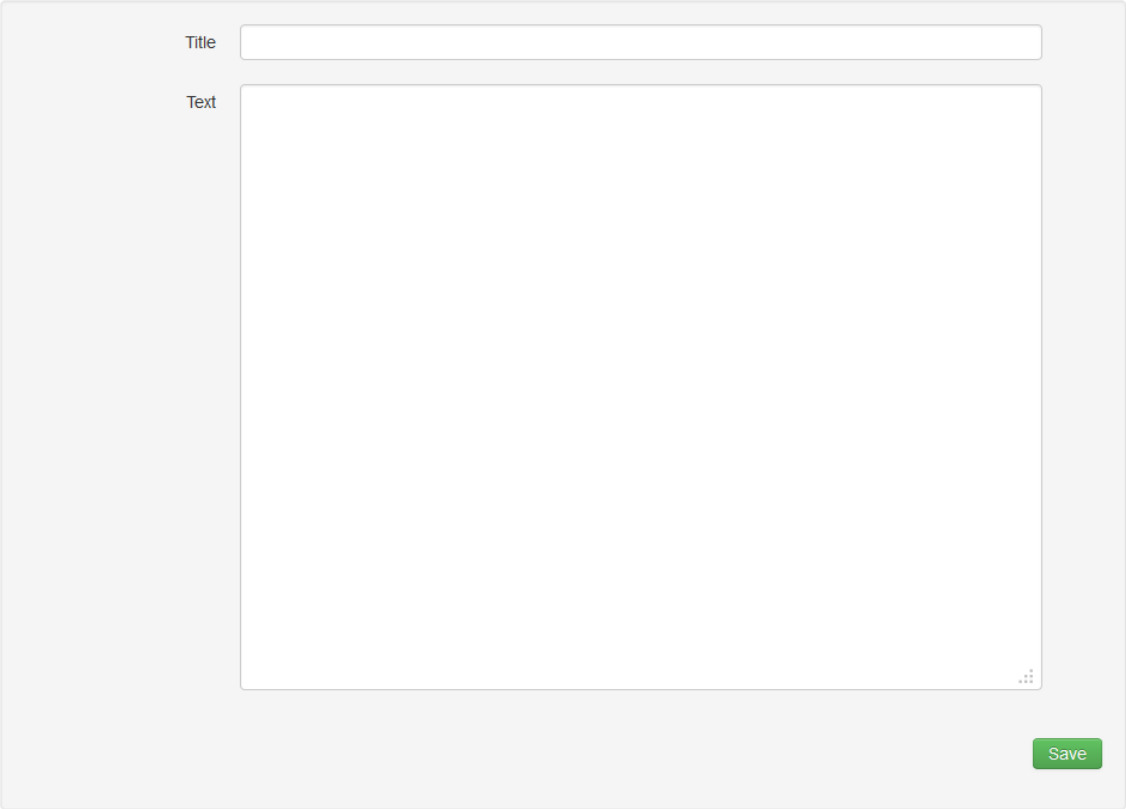


Imagen 13.36. Crear nueva notificación

Cuando terminemos de introducir los datos debemos guardarlos haciendo clic en el botón *Save*. Si todo fue bien se nos redirigirá al panel principal y se nos mostrará un mensaje como el siguiente:

Imagen 13.37. Crear nueva notificación. Mensaje finalización

Ver notificaciones

Si el usuario hotel desea visualizar las notificaciones publicadas por el hotel debe pulsar sobre el icono o nombre *Notifications* de su panel principal. Tras ello, se mostrará la lista con las opciones de modificar o borrar cada una de las notificaciones.

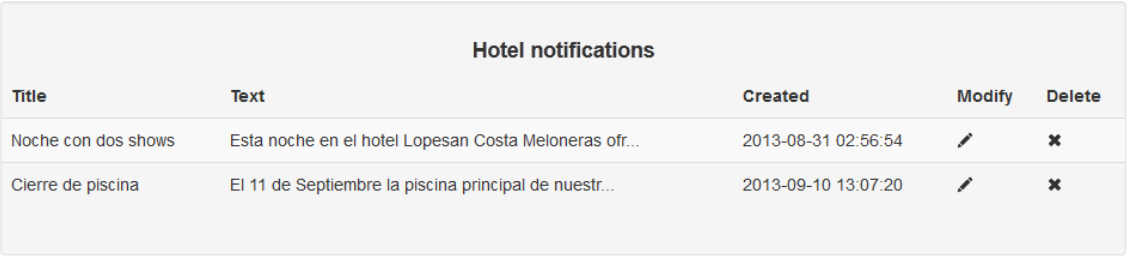


Imagen 13.38. Lista de notificaciones publicadas por el hotel

Borrar una notificación

Para borrar una notificación debemos hacer clic sobre el icono en forma de equis de la fila correspondiente. En ese momento aparecerá en pantalla un cuadro de diálogo para confirmar la eliminación. Pulsando sobre el botón rojo *Delete* la notificación quedará finalmente eliminada.

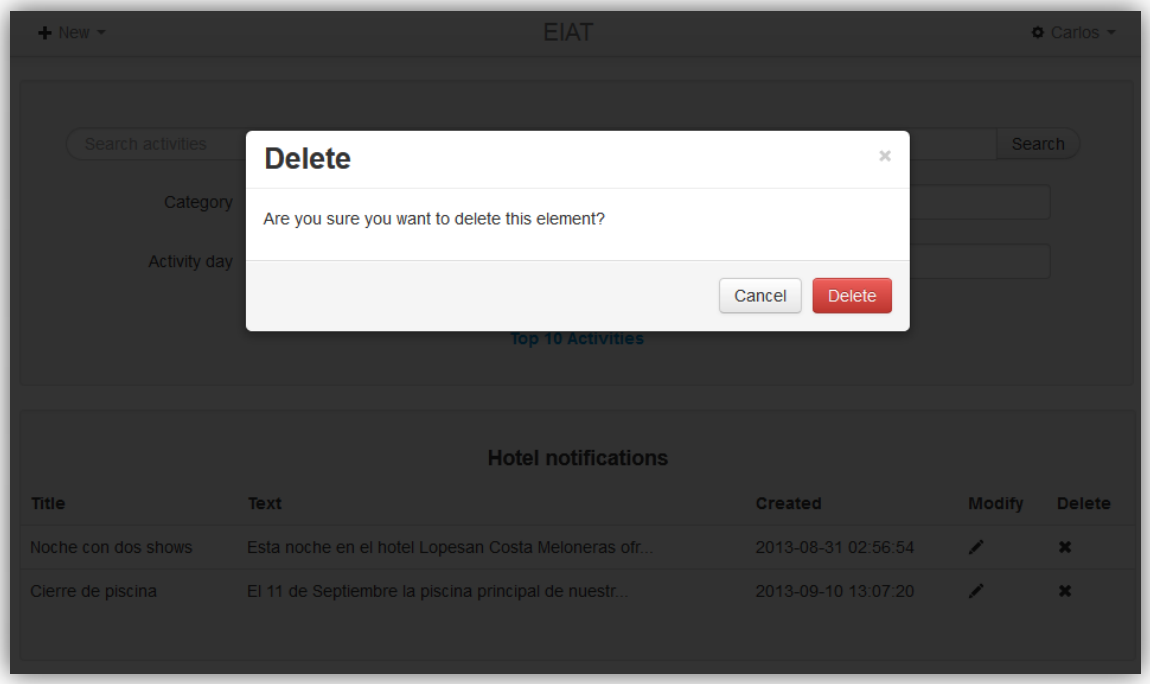


Imagen 13.39. Eliminación de notificación

Modificar una notificación

Cuando se desee modificar una notificación se debe hacer clic sobre el icono de bolígrafo de la fila asociada a la notificación a modificar. Tras realizar esto nos aparecerá un formulario con los datos de la notificación que podremos modificar.

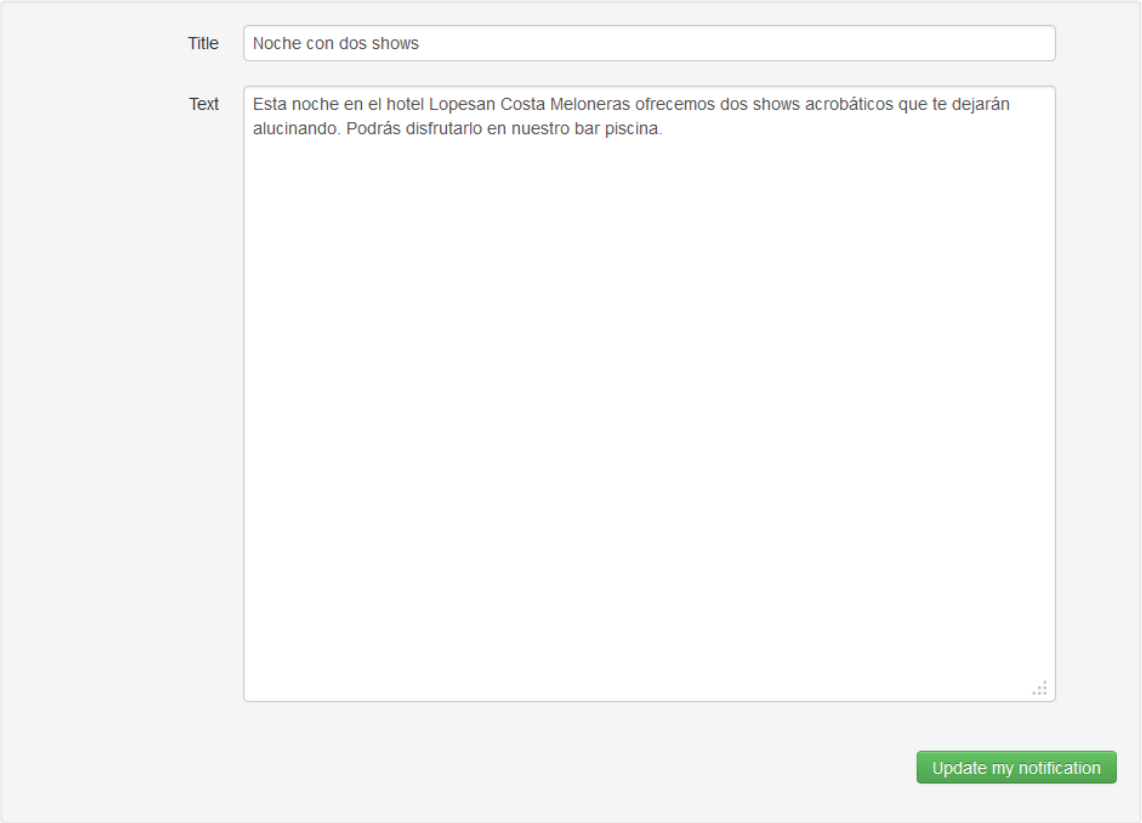


Imagen 13.40. Modificación de una notificación

Cuando finalicemos la edición de la notificación debemos hacer clic sobre *Update my notification* y se nos mostrará el siguiente mensaje:

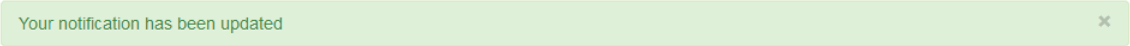


Imagen 13.41. Modificación de una notificación. Mensaje de finalización

Añadir actividad a sugeridas por el hotel

Podremos añadir cualquier actividad de una empresa a nuestra lista de actividades sugeridas. Para ello debemos ver el detalle de la actividad deseada y nos aparecerá un botón que pone *Add to suggestions*. Simplemente pulsando sobre ese botón la actividad ya aparecerá en nuestra lista de actividades sugeridas.

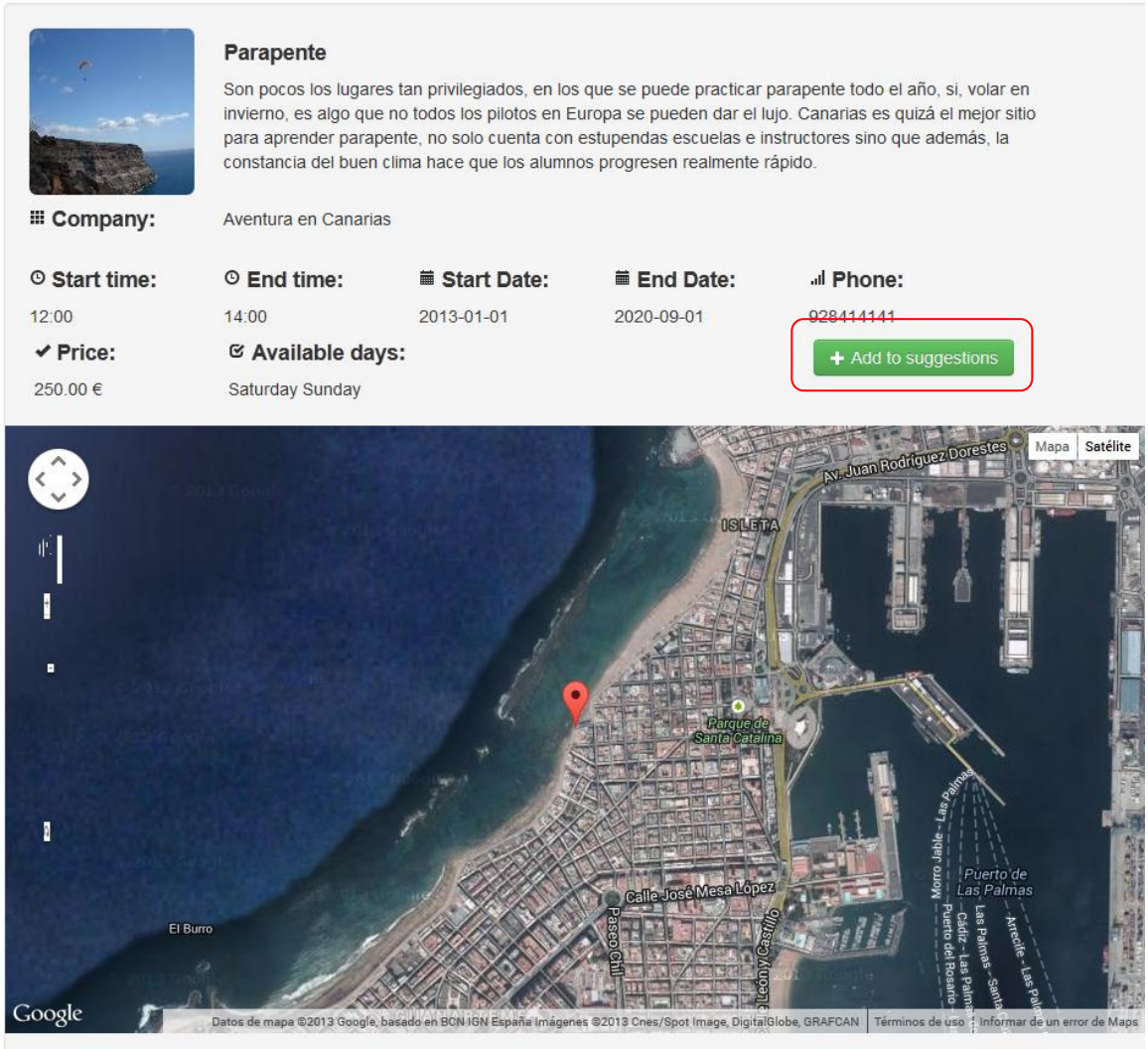


Imagen 13.42. Añadir actividad a la lista de sugeridas

Visualizar actividades sugeridas por el hotel

Para visualizar las actividades sugeridas por el hotel se debe hacer clic sobre el icono de *Suggested activities* del panel principal. A continuación aparecerá el listado con las actividades sugeridas.

Suggested activities

Name	Description	StartDate	EndDate	Price	StartTime	EndTime	Createdat	Delete
Curso Submarinismo y apnea	Son pocos los sitios en Europa donde se puede prac...	2013-01-01 00:00:00	2020-09-01 00:00:00	200.00	12:00:00	17:00:00	2013-08-29 02:41:17	✖
Circuito de talasoterapia 2h	Nuestro Circuito Atlántico está compuesto de: ...	2013-01-01 00:00:00	2020-10-01 00:00:00	20.00	10:00:00	20:00:00	2013-08-29 03:20:18	✖

Imagen 13.43. Listado de actividades sugeridas

Borrar actividad de sugeridas por el hotel

Desde el listado de actividades sugeridas es posible eliminarlas de la lista haciendo clic sobre el icono de equis en la fila correspondiente a la actividad a eliminar. Deberemos confirmar la acción pulsando sobre el botón rojo *Delete*.

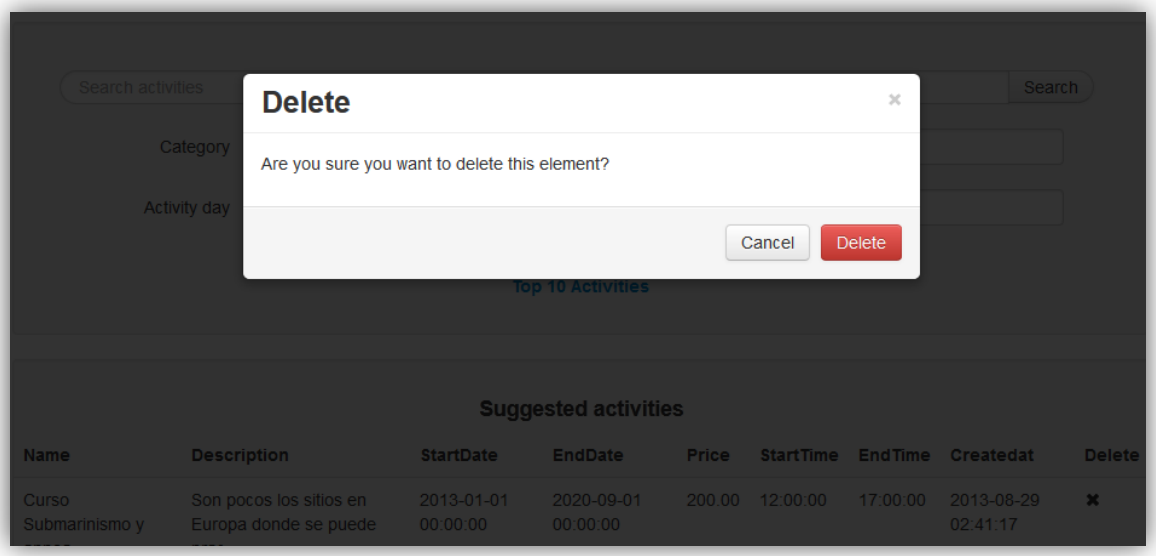


Imagen 13.44. Eliminar actividad sugerida

Rol de hotel administrador

Panel principal

El usuario hotel administrador tiene un completo panel donde aparte de las funcionalidades que puede llevar a cabo un usuario hotel puede además gestionar los usuarios y ver las estadísticas del hotel.

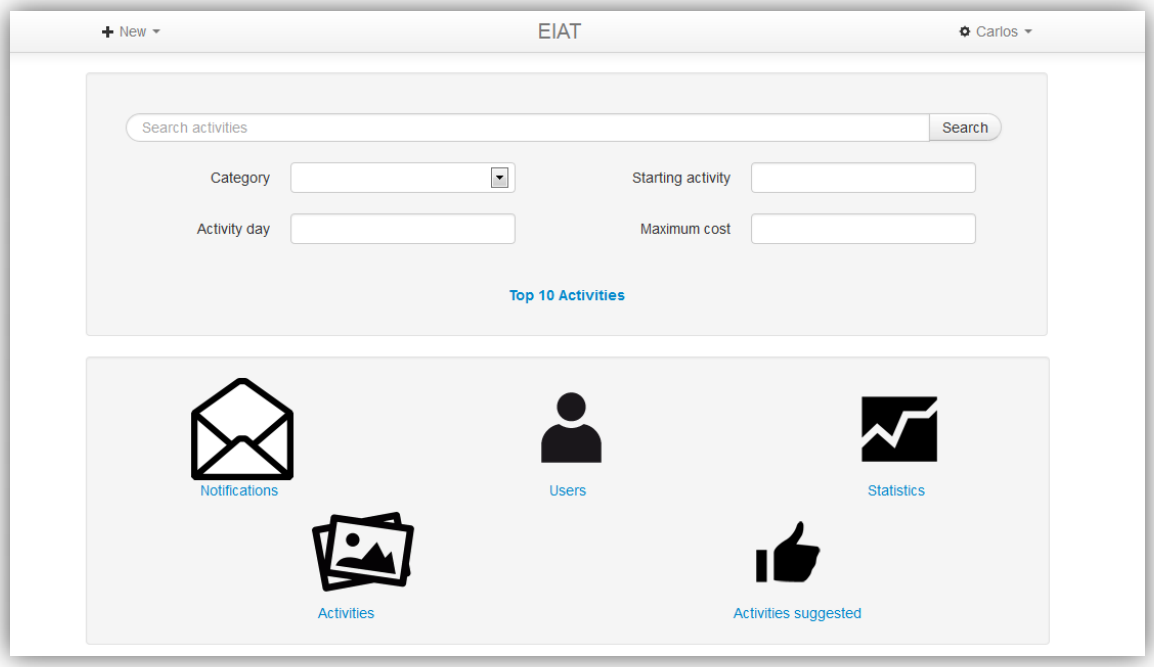


Imagen 13.45. Panel principal del usuario hotel administrador

Menú de acciones

El menú de acciones permite a los usuarios con el rol de hotel administrador crear actividades, usuarios y notificaciones.

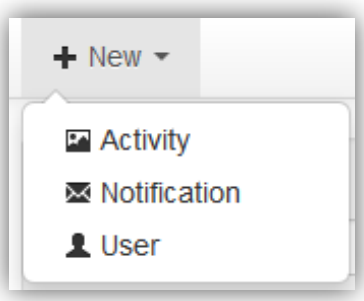


Imagen 13.46. Menú de acciones del usuario hotel administrador

Estadísticas de hotel

El usuario hotel administrador puede visualizar las estadísticas de su empresa haciendo clic sobre el icono o el rótulo *Statistics* de su panel principal. La pantalla de estadísticas muestra varias gráficas. La primera de ellas es un gráfico de porcentajes que representa la distribución de la popularidad entre las actividades del hotel.

Justo a su derecha aparece un resumen donde se indica el número de actividades que posee el hotel, el número de notificaciones publicadas, el número de usuarios en el hotel y el número de veces que sus actividades se encuentran en los favoritos de los usuarios clientes.

El módulo inferior muestra la lista de actividades del hotel. Pulsando sobre cada una de ellas se nos mostrará una gráfica con el *timeline* de visualizaciones y la media de rating de la actividad seleccionada.

Finalmente, el último módulo muestra un *timeline* con el número de uniones de usuarios que el hotel a recibido cada día.

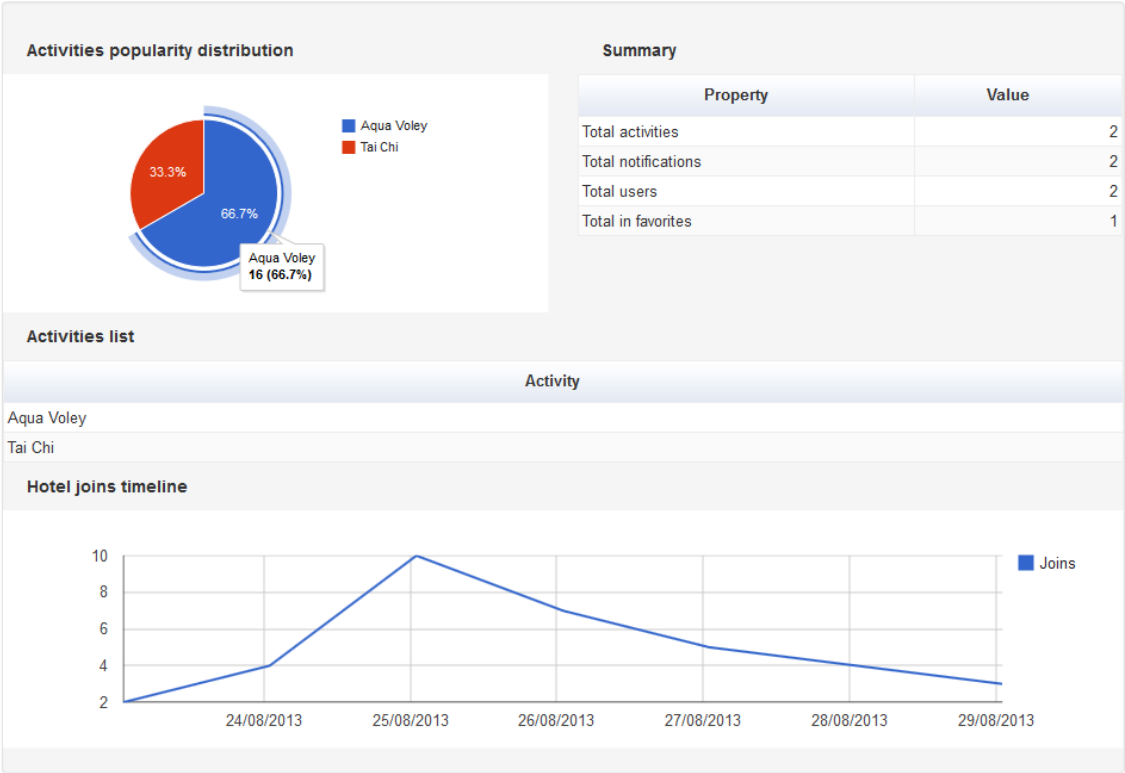


Imagen 13.47. Estadísticas del hotel

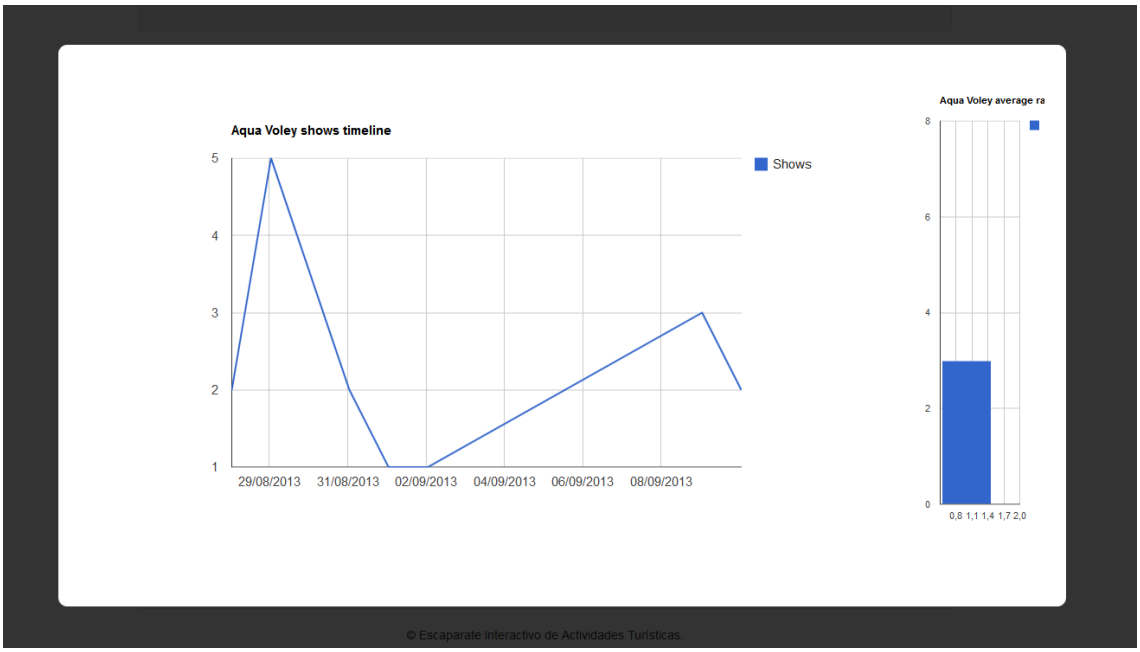


Imagen 13.48. Estadísticas de una actividad del hotel

Rol de empresa

Panel principal

En el panel principal de las empresas es posible gestionar las actividades de la misma. Para su visualización, edición y borrado debemos hacer clic sobre el icono o rótulo *Activities*. Para crear una nueva actividad debemos acudir al menú de acciones del usuario.

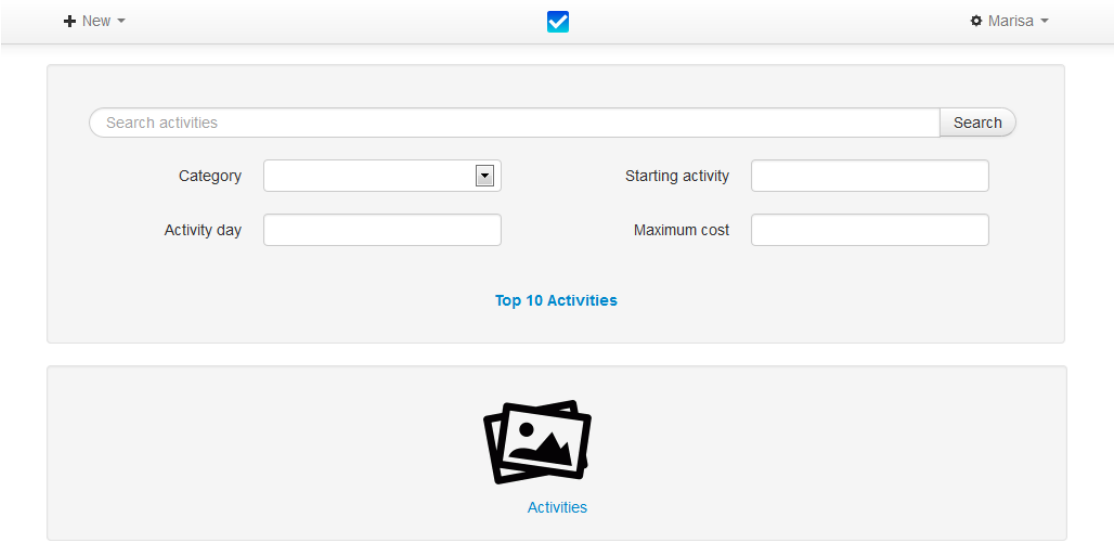


Imagen 13.49. Panel principal del usuario empresa

Menú de acciones

Para crear una nueva actividad el usuario empresa debe acceder a su menú de acciones y hacer clic sobre *New* y a continuación *Activity*.

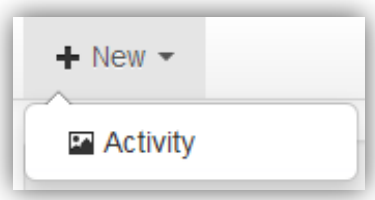


Imagen 13.50. Menú de acciones del usuario empresa

Rol de empresa administrador

Panel principal

El panel principal del un usuario empresa administrador muestra el acceso a las actividades, usuario y estadísticas de la empresa.

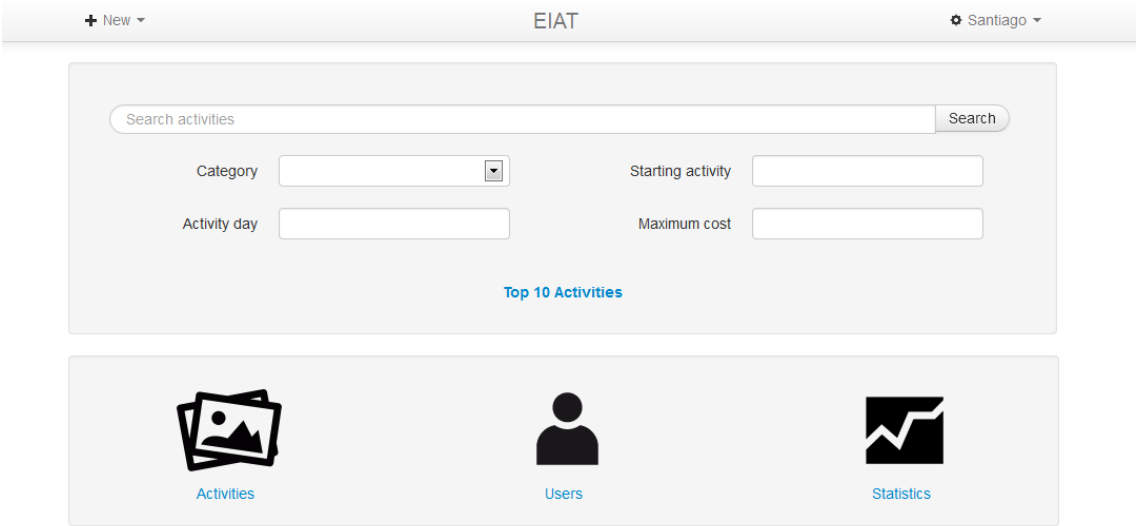


Imagen 13.51. Panel principal del usuario empresa

Menú de acciones

Desde el menú de acciones es posible crear nuevas actividades y usuarios.

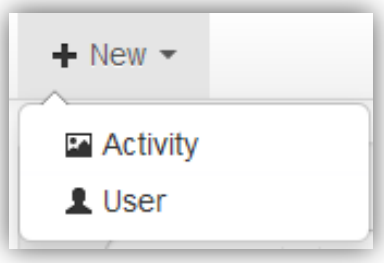


Imagen 13.52. Menú de acciones del usuario empresa administrador

Estadísticas de empresa

El usuario administrador de empresa puede visualizar las estadísticas de la empresa. Estas estadísticas son muy similares a las del hotel. En primer lugar nos encontramos con la gráfica de distribución en porcentajes de popularidad de las actividades de la empresa.

Justo a su derecha aparece una tabla resumen con el número total de actividades de la empresa, el número de veces que sus actividades están siendo sugeridas por hoteles, el número de usuarios y el número de veces que sus actividades están en las listas de favoritos de los usuarios clientes.

En el módulo inferior aparecen las actividades de la empresa, que pinchando sobre ellas se visualiza su *timeline* de visualizaciones y la media de las votaciones de los usuarios clientes.

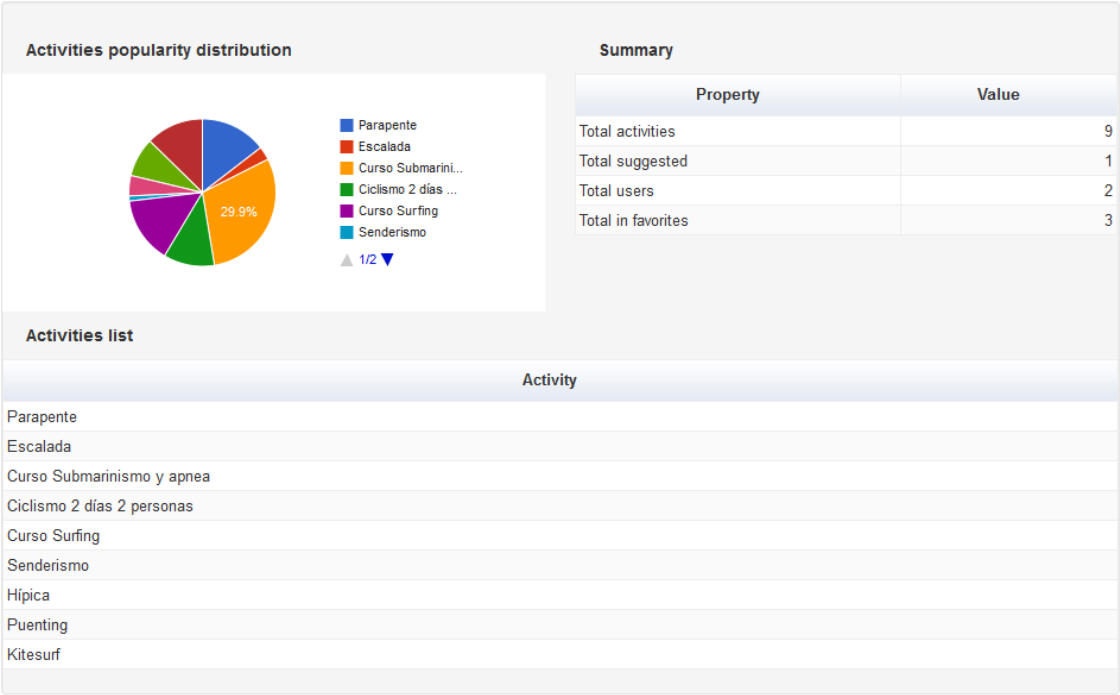


Imagen 13.53. Estadísticas del usuario empresa administrador

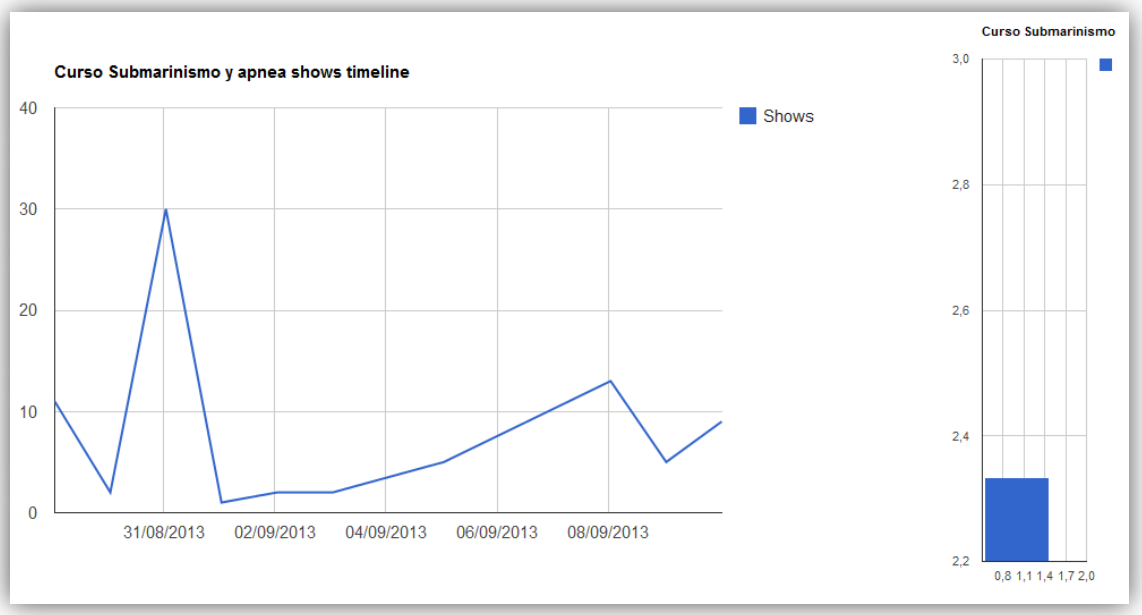


Imagen 13.54. Estadística de una actividad

Rol de administrador local

Crear usuario

Cuando un usuario hotel o empresa administrador desee crear un usuario de su negocio debe hacer clic sobre *User* de su menú de acciones. En ese momento le aparecerá un formulario como el siguiente. En el caso de que se esté administrando una empresa las opciones de roles serán *Company user* y *Company admin user*, y en el caso de que sea un hotel aparecerán *Hotel user* y *Hotel admin user*.

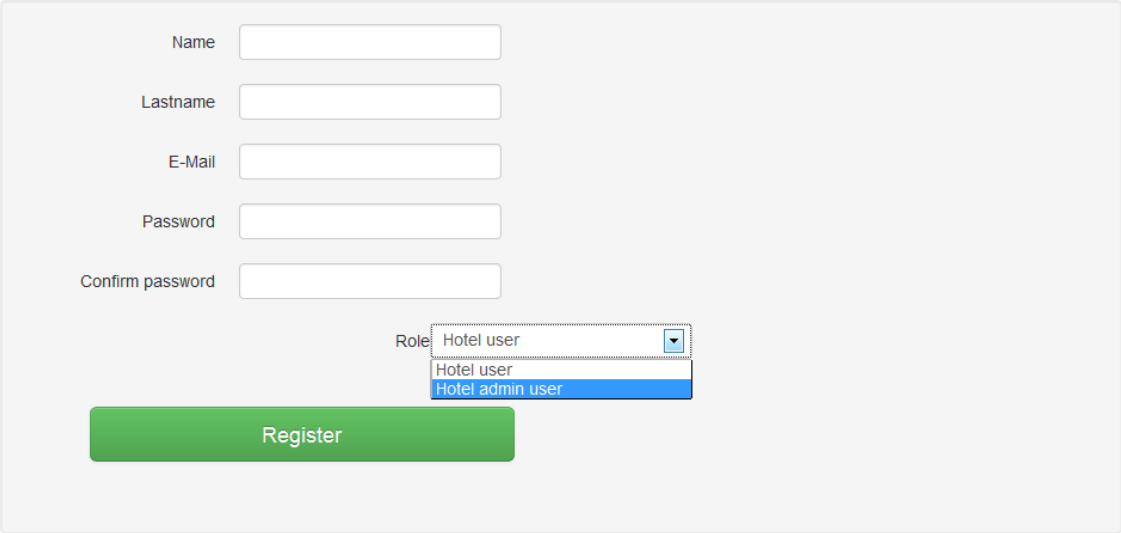


Imagen 13.55. Crear nuevo usuario de hotel

Una vez que se hayan rellenado todos los campos y seleccionado el rol del usuario sólo es necesario pulsar sobre el botón verde *Register*.

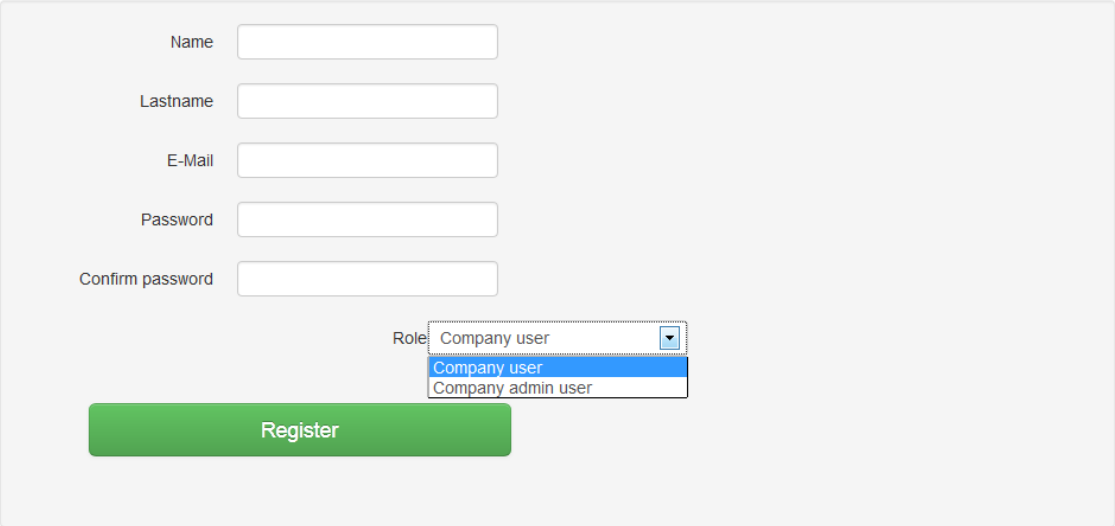


Imagen 13.56. Crear nuevo usuario empresa

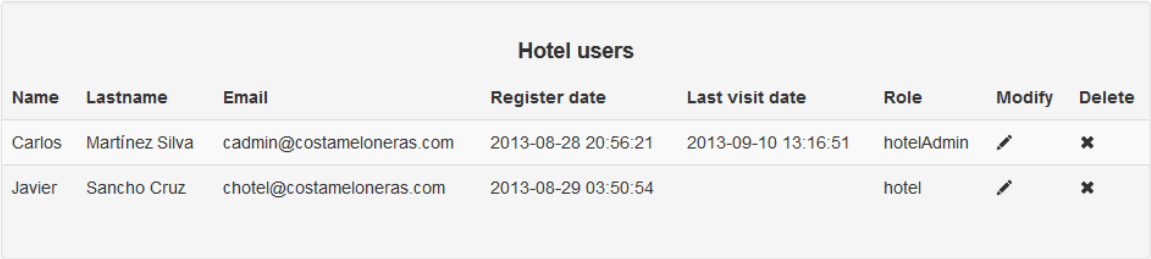
Si el proceso de registro se completa sin errores aparecerá el siguiente mensaje:

Imagen 13.57. Crear un nuevo usuario de la compañía. Mensaje de confirmación

Visualizar usuarios

Para visualizar los usuario que pertenecen a la compañía se debe hacer clic sobre el icono o rótulo *Users* del panel principal.

A los usuarios hotel administradores se les mostrará un listado similar al siguiente:



Hotel users

Name	Lastname	Email	Register date	Last visit date	Role	Modify	Delete
Carlos	Martínez Silva	cadmin@costameloneras.com	2013-08-28 20:56:21	2013-09-10 13:16:51	hotelAdmin		
Javier	Sancho Cruz	chotel@costameloneras.com	2013-08-29 03:50:54		hotel		

Imagen 13.58. Lista de usuarios de un hotel

En el caso de los usuarios empresa administradores aparecerá un panel análogo.

Company users

Name	Lastname	Email	Register date	Last visit date	Role	Modify	Delete
Santiago	Rodriguez del Pino	aadmin@aventuracanarias.com	2013-08-29 02:26:40	2013-09-12 00:39:29	companyAdmin		
Marisa	López Vega	acompany@aventuracanarias.com	2013-08-29 03:10:58	2013-09-10 15:24:49	company		

Imagen 13.59. Lista de usuarios de una empresa

Modificar un usuario

Para modificar un usuario se debe hacer clic en el icono de lápiz correspondiente a la fila del usuario que se quiere modificar de la lista de usuarios de la compañía. En ese momento nos aparecerá un formulario con los datos del usuario que podremos modificar y guardar.

Name: Carlos
Lastname: Martínez Silva
E-Mail: cadmin@costameloneras.com
Password
Confirm password
Role: Hotel admin user
Update user

Imagen 13.60. Actualización de un usuario de la compañía

Si el proceso de actualización de datos del usuario ha sido llevado a cabo correctamente se mostrará el siguiente mensaje:

Your user has been updated

Imagen 13.61. Actualización de un usuario de la compañía. Mensaje de confirmación

Borrar un usuario

Para borrar un usuario se debe hacer clic sobre el icono de equis a la fila correspondiente al usuario a eliminar de la lista de usuarios de la compañía. Acto

seguido se mostrará un cuadró de confirmación de la eliminación. Para finalizar debemos pulsar sobre el botón rojo *Delete*. El usuario se suprimirá del sistema y desaparecerá de la lista.

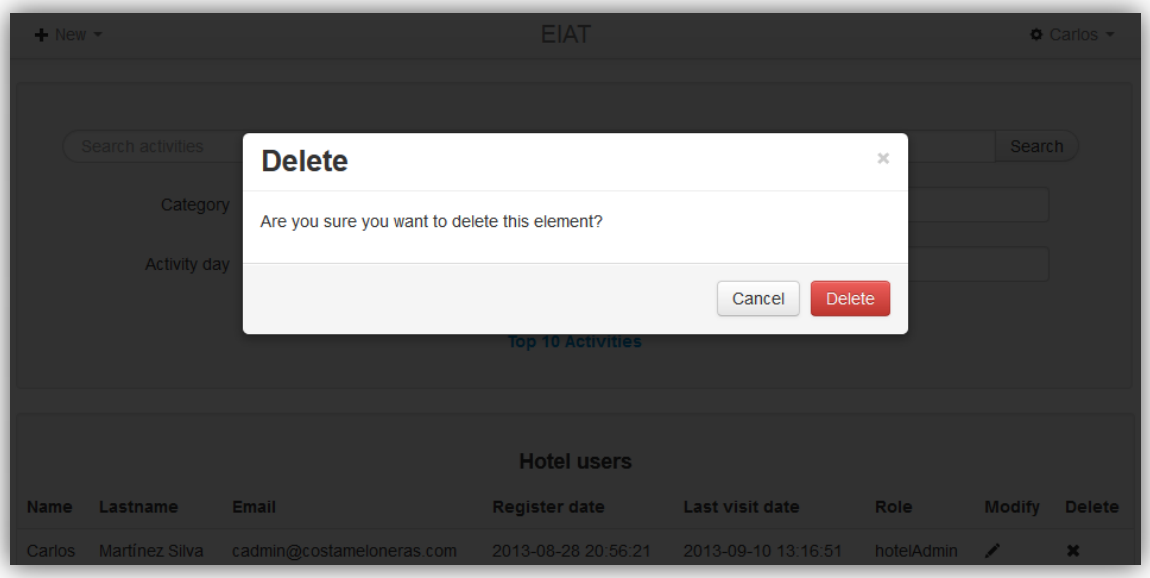


Imagen 13.62. Eliminación de un usuario

Rol tipo empresa

El rol de tipo empresa comprendía las siguientes cuatro funcionalidades:

- Crear actividad
- Ver actividades
- Modificar actividad
- Borrar actividad

Crear una actividad

Para crear una actividad se debe hacer clic sobre *Activity* del menú de acciones de la compañía. A continuación aparecerá un formulario para rellenar todos los datos de la actividad.

Será necesario rellenar todos los campos para que la actividad quede registrada. Para facilitar el formato de completado de los campos de fechas y horas se activarán los *datepicker* y *timepicker*.

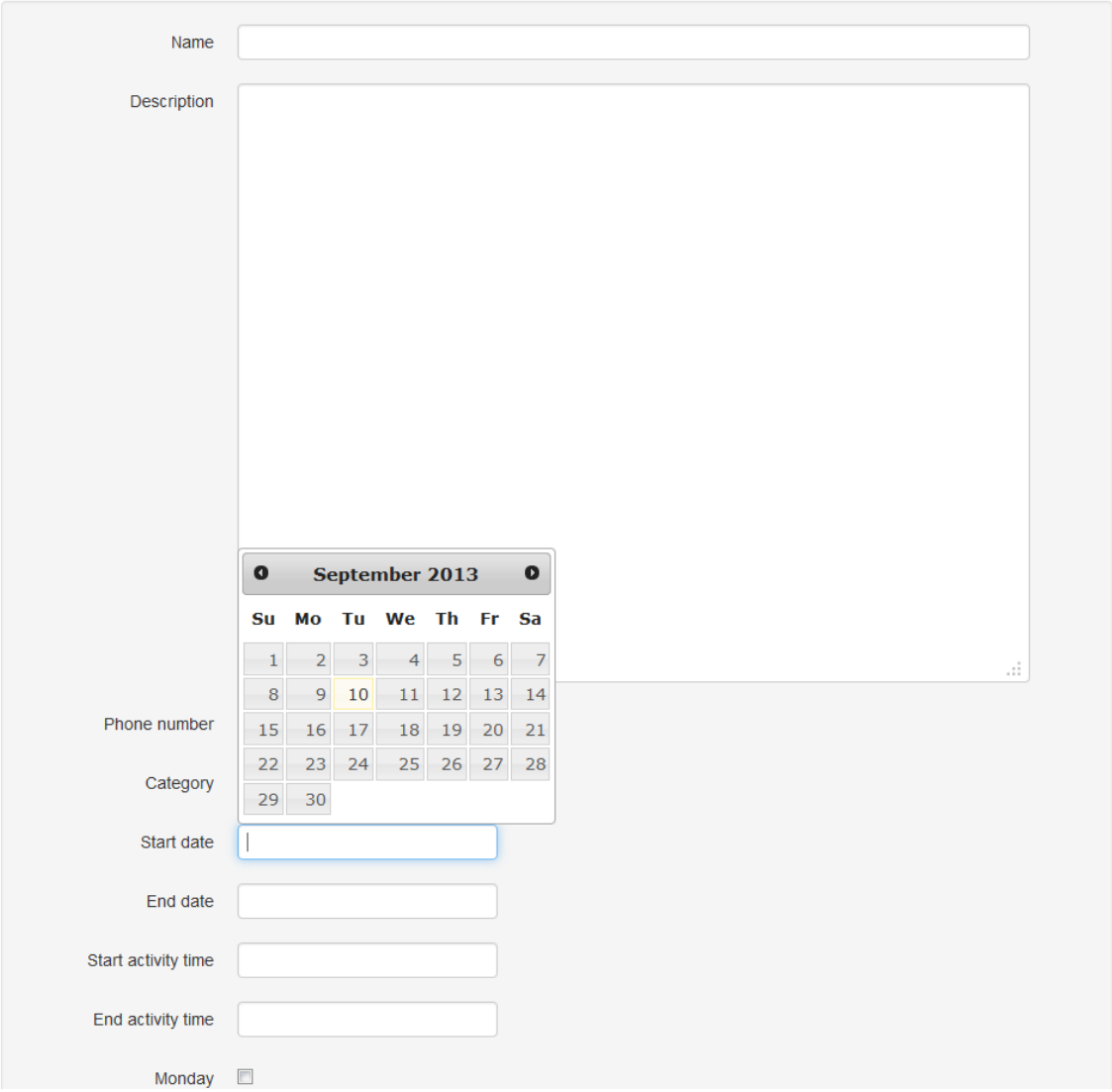


Imagen 13.63. Crear actividad (arriba)

Para localizar la actividad debemos introducir la dirección en el campo *Address of activity* y pulsar sobre el botón azul *Localize activity*. Automáticamente el mapa se centrará sobre la dirección proporcionada y los campos de coordenadas quedarán rellenados.

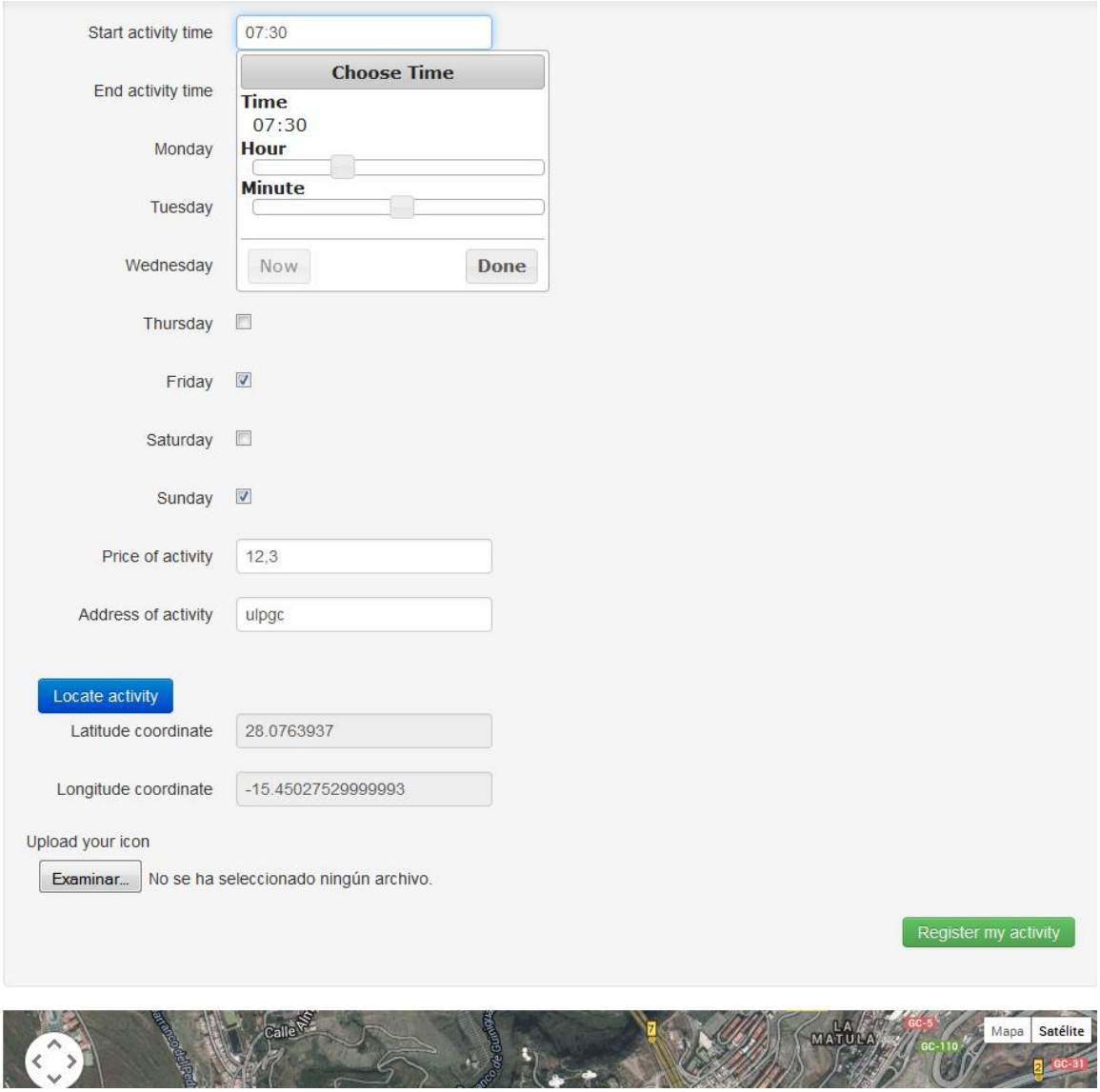


Imagen 13.64. Crear actividad (abajo)

Para finalizar la creación de la actividad debemos hacer clic sobre *Register my activity* y si todo fue bien se mostrará el siguiente mensaje:

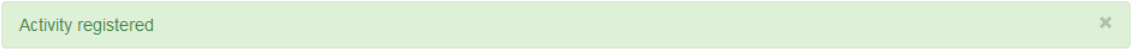


Imagen 13.65. Crear actividad. Mensaje de confirmación

En caso contrario reaparecerá el formulario con los datos introducidos y los mensajes de error debajo de los campos donde se detectaron las incorrecciones.

Ver actividades

Para visualizar las actividades de la compañía debe hacer clic sobre el icono o el rótulo *Activities* del panel principal. Las actividades se muestran en forma de tabla con muchos de sus datos.

Hotel activities

Name	Description	Start date	End date	Price	Start time	End time	Created	Modify	Delete
Aqua Voley	Disfrute de partidos de voley en nuestra enorme pi...	2000-12-01 00:00:00	2014-09-01 00:00:00	0.00	12:30:00	13:00:00	2013-08-28 21:11:36		
Tai Chi	Sesiones de Tai Chi impartidas por nuestros profes...	2013-01-01 00:00:00	2014-09-01 00:00:00	0.00	19:00:00	20:00:00	2013-08-28 21:24:32		

Imagen 13.66. Lista de actividades de un hotel

El formato de la lista de actividades de los usuarios hotel y empresas son muy similares.

Company activities

Name	Description	Start date	End date	Price	Start time	End time	Created	Modify	Delete
Parapente	Son pocos los lugares tan privilegiados, en los qu...	2013-01-01 00:00:00	2020-09-01 00:00:00	250.00	12:00:00	14:00:00	2013-08-29 02:32:19		
Escalada	Por su geología las islas canarias son el sitio pe...	2013-01-01 00:00:00	2013-12-01 00:00:00	60.00	10:00:00	20:00:00	2013-08-29 02:37:59		
Curso Submarinismo y apnea	Son pocos los sitios en Europa donde se puede prac...	2013-01-01 00:00:00	2020-09-01 00:00:00	200.00	12:00:00	17:00:00	2013-08-29 02:41:17		
Ciclismo 2 días 2 personas	Practicando el ciclismo de montaña se puede conoce...	2013-01-01 00:00:00	2020-09-01 00:00:00	85.00	10:00:00	18:00:00	2013-08-29 02:44:24		
Curso Surfing	Los mejores surfers colaboran con Aventura en Cana...	2013-01-01 00:00:00	2013-09-01 00:00:00	143.00	10:00:00	20:00:00	2013-08-29 02:48:14		
Senderismo	Desde el Refugio de Pilar subimos hasta Las Desead...	2013-01-01 00:00:00	2020-09-01 00:00:00	10.00	10:00:00	18:00:00	2013-08-29 02:52:22		
Hípica	Montar a caballo es una experiencia sin igual, rel...	2013-01-01 00:00:00	2020-09-01 00:00:00	40.00	17:00:00	19:00:00	2013-08-29 02:57:13		
Puenting	El puenting es una de las actividades mas atractiv...	2013-03-01 00:00:00	2013-09-01 00:00:00	95.00	12:00:00	14:00:00	2013-08-29 03:04:59		
Kitesurf	Los Alisios son los vientos del Nordeste que sopla...	2013-03-01 00:00:00	2013-09-01 00:00:00	140.00	10:00:00	20:00:00	2013-08-29 03:07:52		

Imagen 13.67. Lista de actividades de una empresa

Modificar una actividad

Para modificar una actividad se debe pulsar sobre el icono de lápiz de la fila correspondiente a la actividad a modificar. La pantalla que se mostrará corresponde a un formulario de registro de actividad pero con los datos rellenados con la actividad seleccionada.

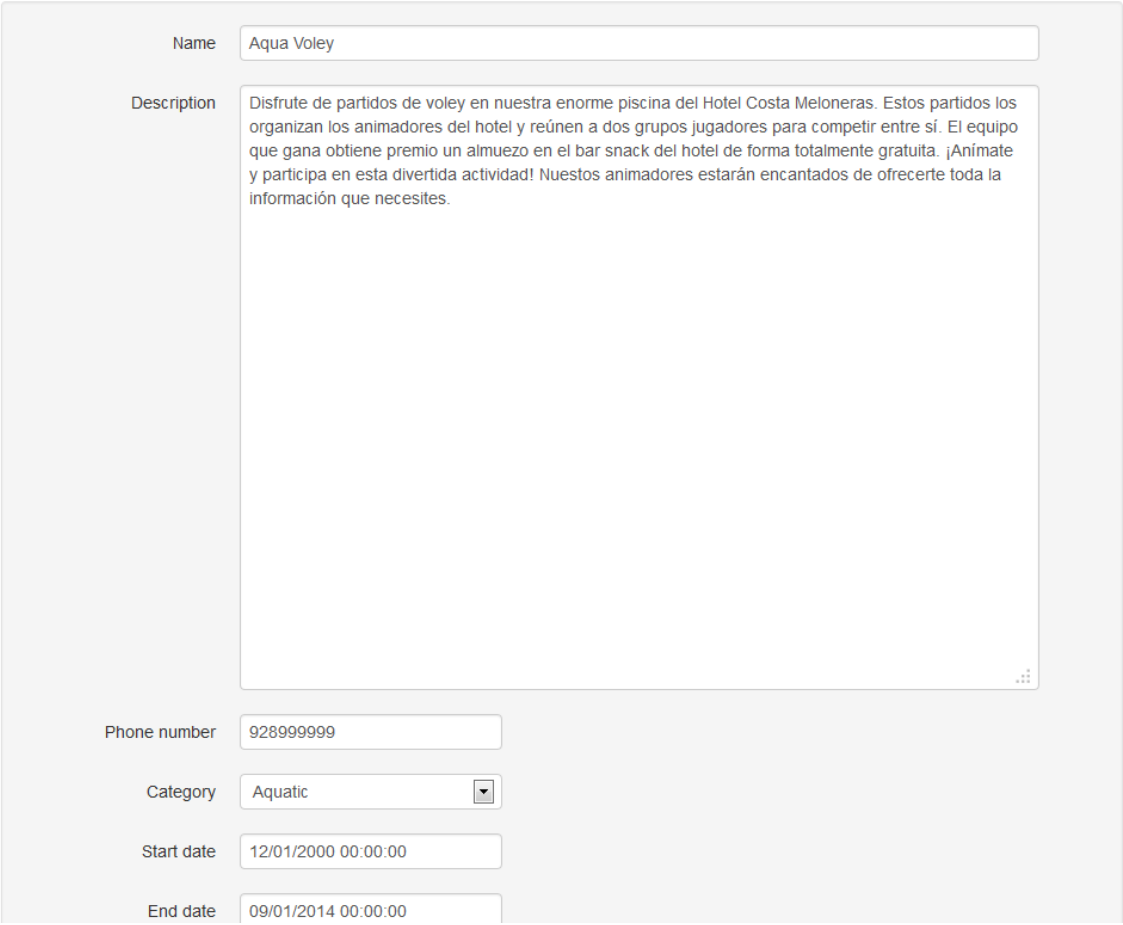


Imagen 13.68. Modificar una actividad

Cuando finalicemos la edición debemos pulsar sobre *Update my activity* y si las modificaciones fueron correctas se mostrará el siguiente mensaje:

Imagen 13.69. Modificar una actividad. Mensaje de confirmación

Borrado de una actividad

Para realizar la eliminación de una actividad debemos pulsar sobre el icono en forma de equis de la fila correspondiente a la actividad a eliminar. En ese

momento nos aparecerá un cuadro de diálogo que nos pedirá la confirmación de la eliminación.

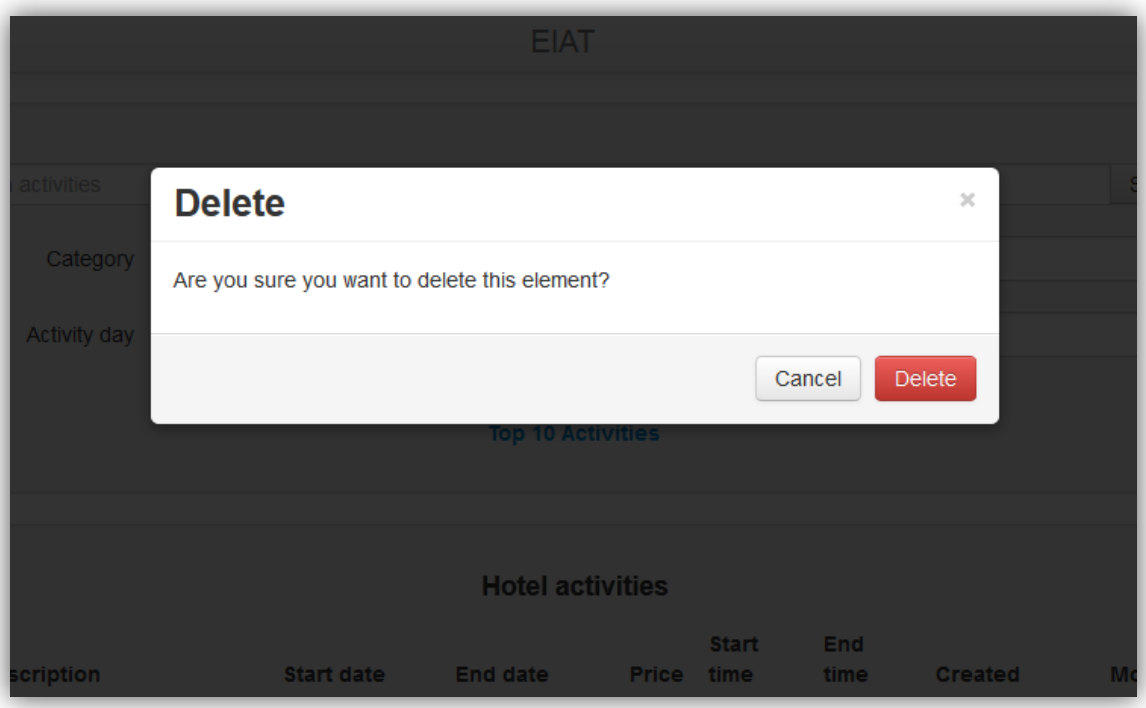


Imagen 13.70. Eliminar una actividad

Para confirmar la eliminación debemos pulsar sobre el botón rojo *Delete* y la actividad quedará definitivamente eliminada.

Rol de administrador local

Menú de usuario de compañías

El menú de usuario de una compañía es muy similar al menú de usuario de un cliente pero en vez de tener unas preferencias tiene un perfil de la empresa o el hotel según corresponda.

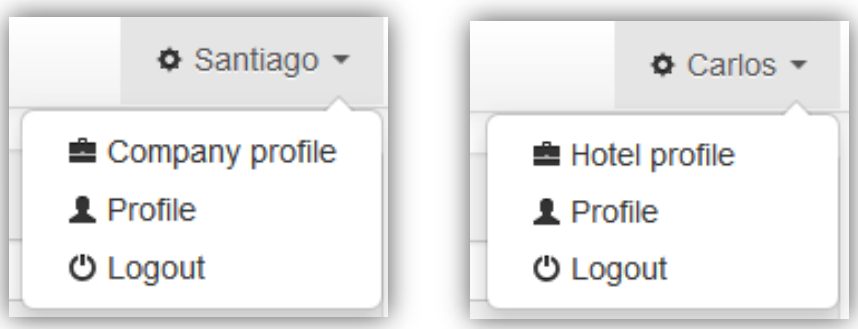


Imagen 13.71. Menú de usuario de una empresa y un hotel

Ver y modificar el perfil de la compañía

Para ver y poder modificar el perfil de la compañía es necesario pulsar sobre el botón *Company profile* del menú de usuario. A continuación aparecerá un formulario con los datos de la compañía que podremos modificar y guardar.

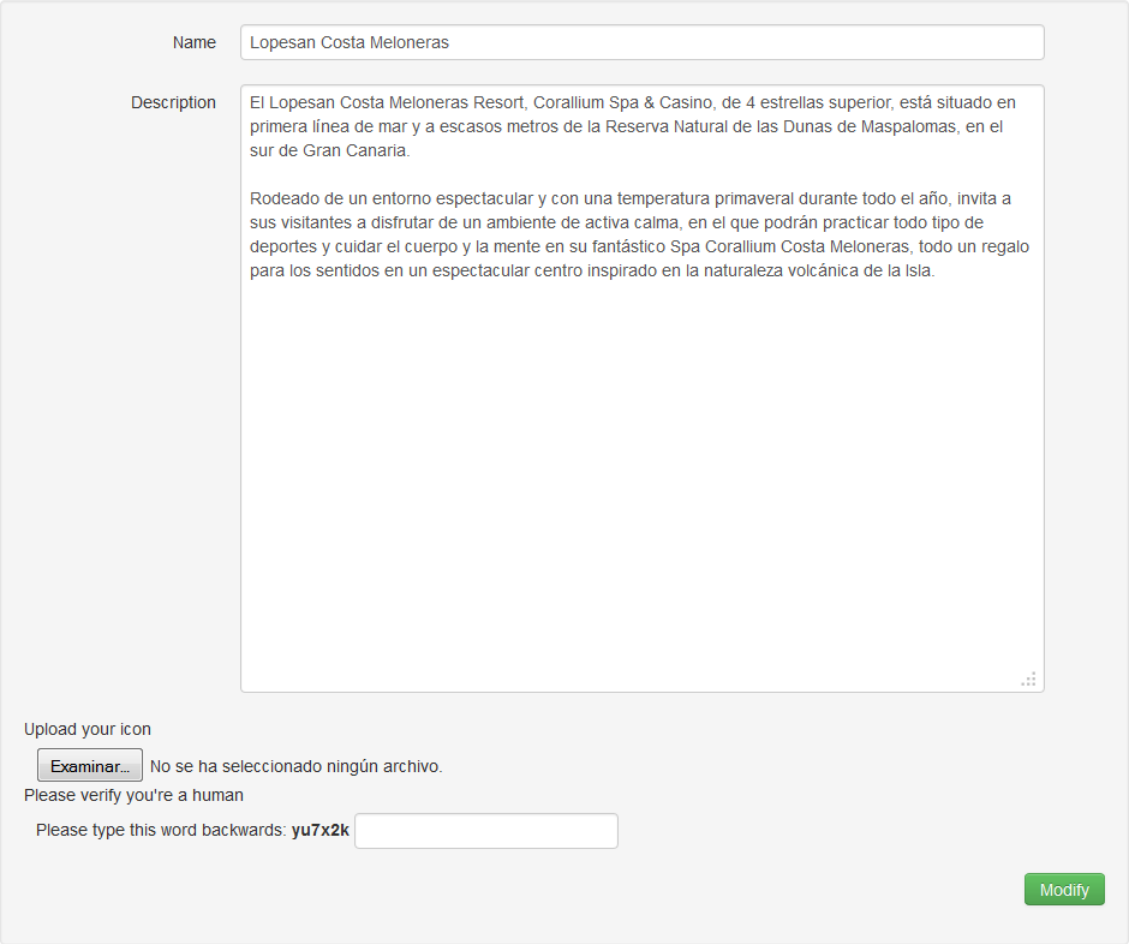


Imagen 13.72. Actualizar perfil de la compañía

En el caso de que algunos o todos los datos sean incorrectos se nos mostrará el formulario con este mensaje:

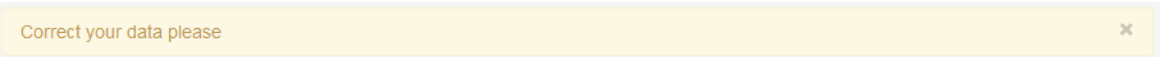


Imagen 13.73. Actualizar perfil de la compañía. Mensaje de error

Por el contrario, si se actualizaron correctamente los datos de la empresa se mostrará este otro mensaje:

Imagen 13.74. Actualizar perfil de la compañía. Mensaje de confirmación

Anexo C. Instalación y configuración de Karotz.

Para realizar la instalación y configuración de Karotz debemos seguir las instrucciones de la página oficial de Karotz:
http://www.karotz.com/plug/download

1. Seleccionar si Karotz se va a conectar mediante cable o Wi-Fi.
2. Rellenar los campos de configuración de la red. En caso de que sea por Wi-Fi y la red no tenga contraseña debemos introducir una cualquiera para continuar el proceso y más adelante modificar los ficheros de configuración.
3. Descargar los ficheros de configuración.
4. Copiar los ficheros del archivo comprimido descargado en la raíz de un USB. En el caso de que fuera conexión Wi-Fi y no tuviera contraseña deberemos editar el fichero "network.conf" y borrar la falsa contraseña que pusimos (aparece en el campo "pwd").
5. Si Karotz no está apagado, apagarlo y desenchufarlo. Insertar el USB en Karotz.
6. Enchuchar a Karotz y encenderlo.
7. Karotz se encenderá y moverá sus orejas. Inicialmente tendrá el color rojo que significa que está iniciándose.
8. A continuación pasará al color naranja que indica que el nuevo sistema está siendo instalado.
9. Finalmente, si todo fue bien, se quedará parpadeando en color verde.

Anexo D. Manual de usuario para Karotz.

1. Encender el aparato.

Debemos desplazar la rueda de encendido y volumen para encender a Karotz.

Imagen 13.75. Encendido de Karotz

2. Arrancar la aplicación.

Para iniciar la aplicación debemos pulsar sobre su botón o si tenemos un nanoztag o un flatnanoz asociado a la aplicación frotárselo por la zona de la nariz.

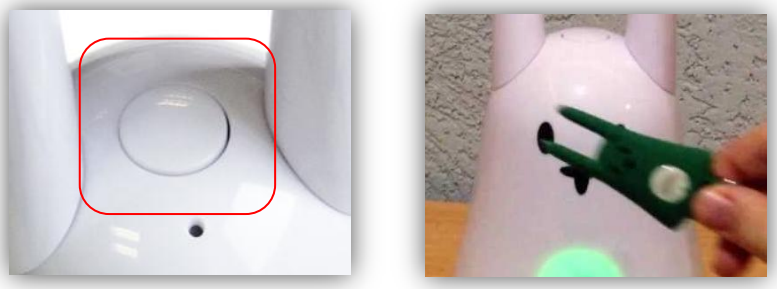


Imagen 13.76. Iniciar la aplicación en Karotz

3. Escuchar las actividades más populares.

Cuando arranquemos la aplicación de Karotz nos preguntará qué deseamos hacer. En ese punto tenemos que decirle que queremos las "top 10 activities" o "top 10". Automáticamente Karotz nos irá diciendo el nombre de cada una de las actividades de esa lista. Podremos escuchar el detalle de cualquiera de ellas con tan sólo decir el nombre de la actividad.

4. Pedir una recomendación.

Cuando arranquemos la aplicación de Karotz nos preguntará qué deseamos hacer. En ese punto tenemos que decirle que queremos "get a recommendation". A continuación Karotz nos dirá las categorías existentes para que elijamos una y finalmente nos preguntará si la queremos para el mismo día o para el día siguiente. Con esos datos Karotz nos dará una recomendación.

Anexo E. Manual de instalación de la aplicación para Android.

Para instalar la aplicación cliente del sistema de recomendación se deben seguir los siguientes pasos:

1. Abrir la aplicación Google Play o Play Store desde nuestro dispositivo Android y buscar "EIAT PFC DNM".

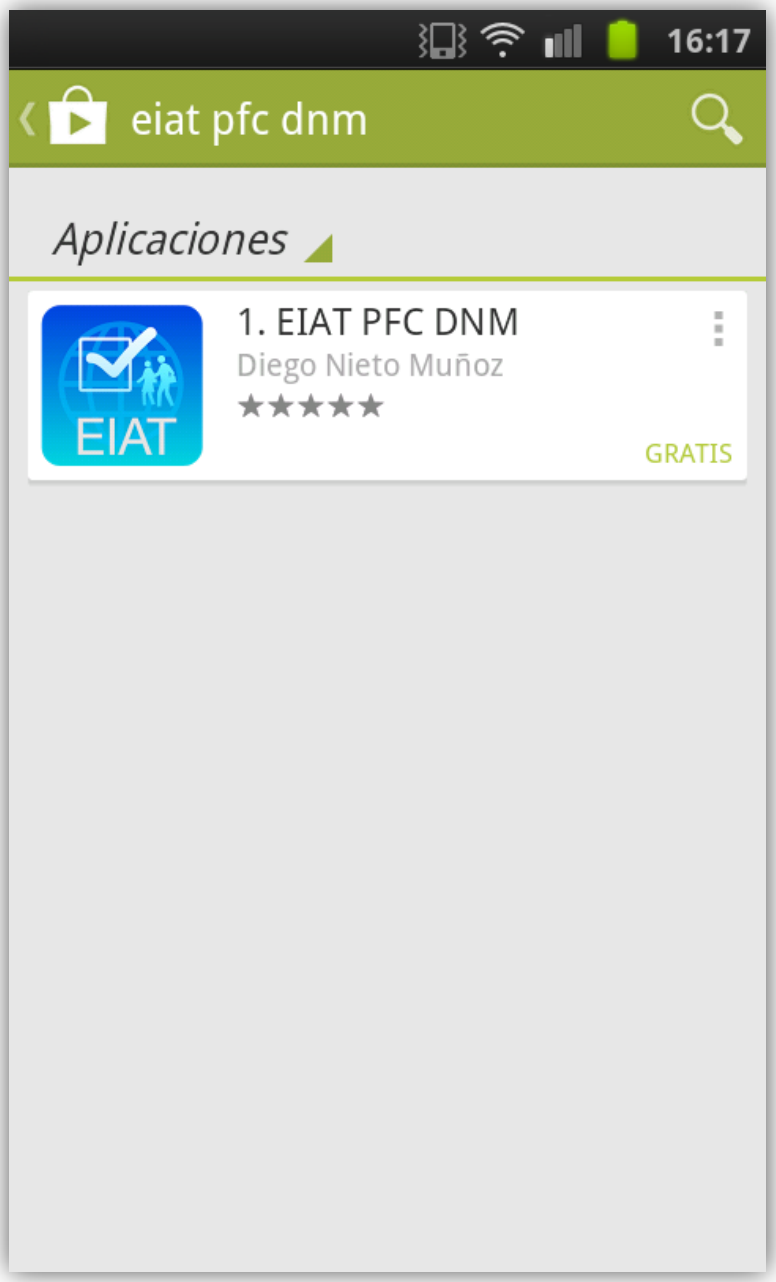


Imagen 13.77. Búsqueda de EIAT en Google Play

2. Abrir el primer ítem que aparece en la lista y pulsar sobre el botón Instalar.

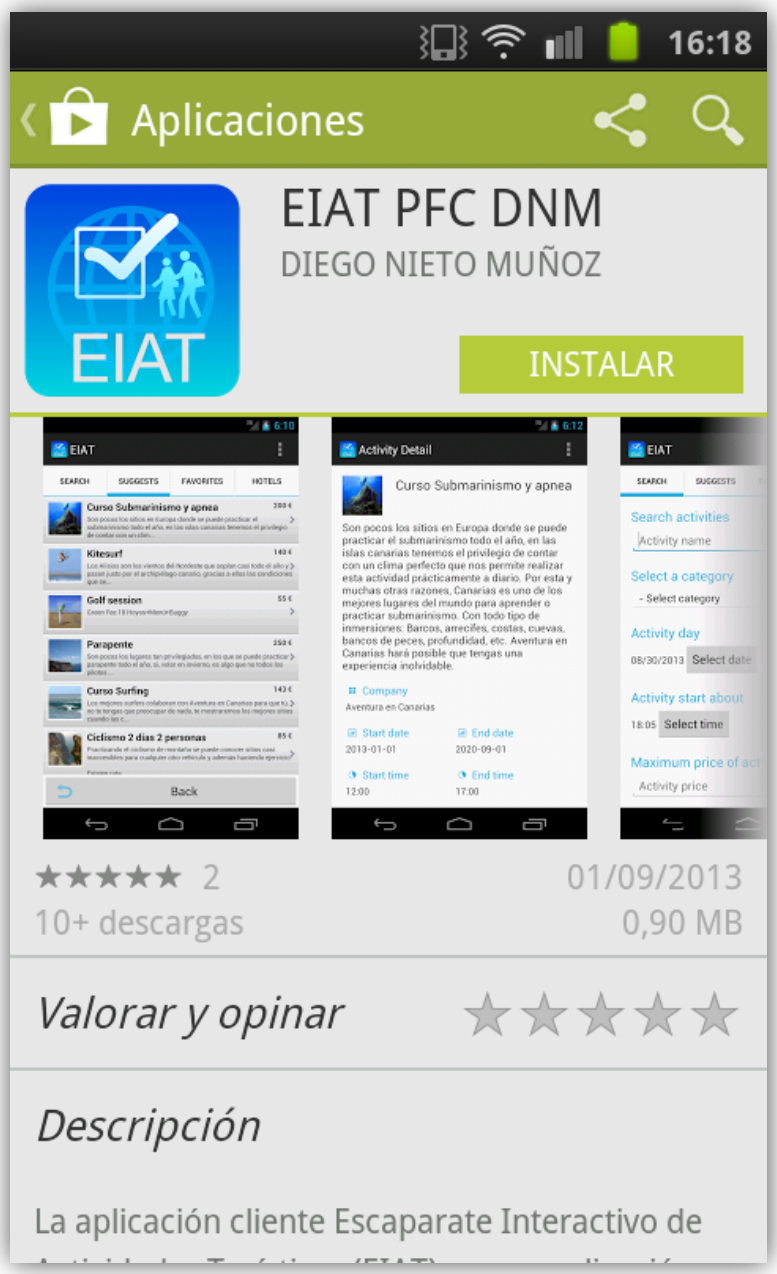


Imagen 13.78. Página de EIAT en Google Play

3. Aceptar los permisos de la aplicación.

Imagen 13.79. Instalación de EIAT en el dispositivo Android

4. Esperar a que se instale. Una vez completada la instalación ya podremos utilizarla.

Anexo F. Manual de usuario de la aplicación para Android.

Inicio de sesión y registro

Para iniciar sesión en la aplicación debemos introducir el correo y la contraseña con los que nos hayamos registrado como cliente en el sistema de recomendación y pulsar sobre *Sign in*. Las siguientes veces que abramos la aplicación iniciará sesión automáticamente con las credenciales introducidas anteriormente.

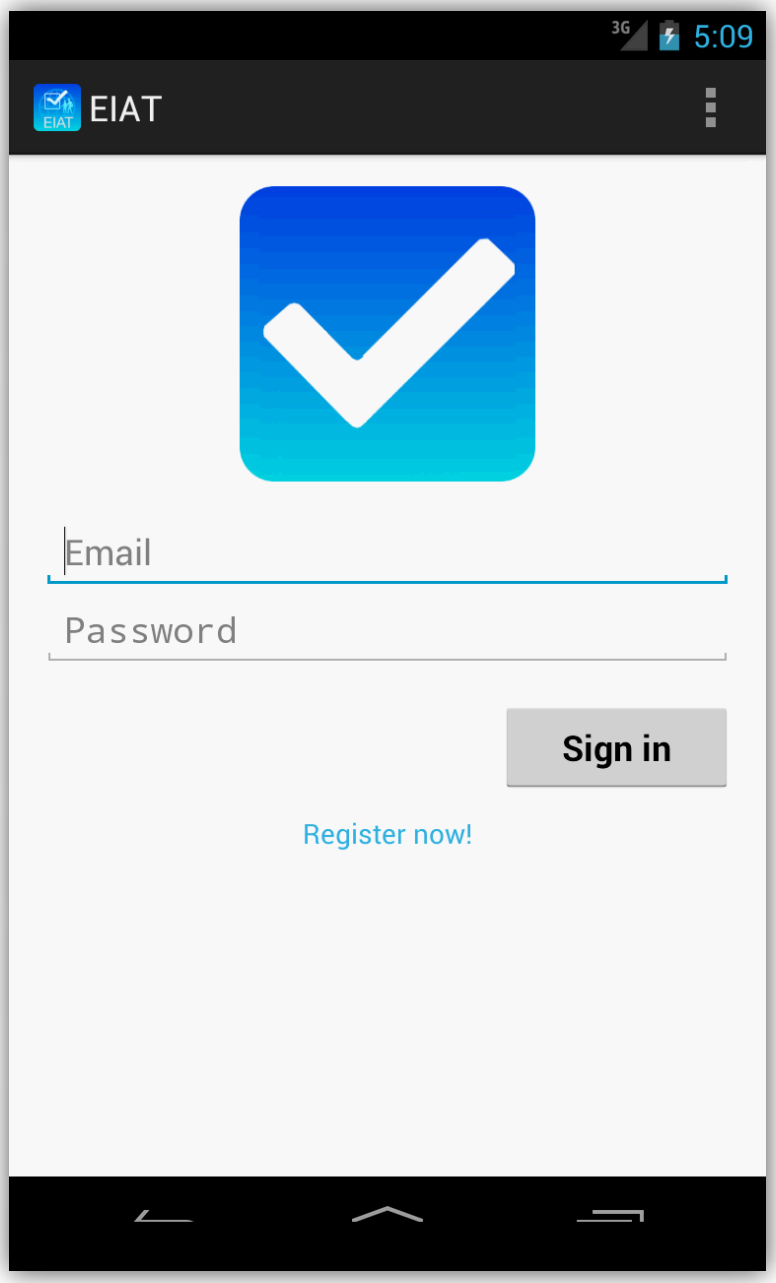


Imagen 13.80. Android. Pantalla de login

En el caso de que todavía no dispongamos de una cuenta podremos registrarnos en el sistema pulsando sobre *Register now!*.

Panel principal y pestaña de búsqueda

El panel principal de la aplicación cliente para Android está dividido en cuatro pestañas: Search, For you, Favs y Hotel. La primera permite realizar búsquedas parametrizables de actividades, la segunda muestra las opciones de recibir una recomendación personalizada o ver las *Top 10 activities*, la tercera la lista de actividades favoritas y la cuarta un buscador de hoteles si no estamos unidos a un hotel o las opciones del hotel al que estamos unido.

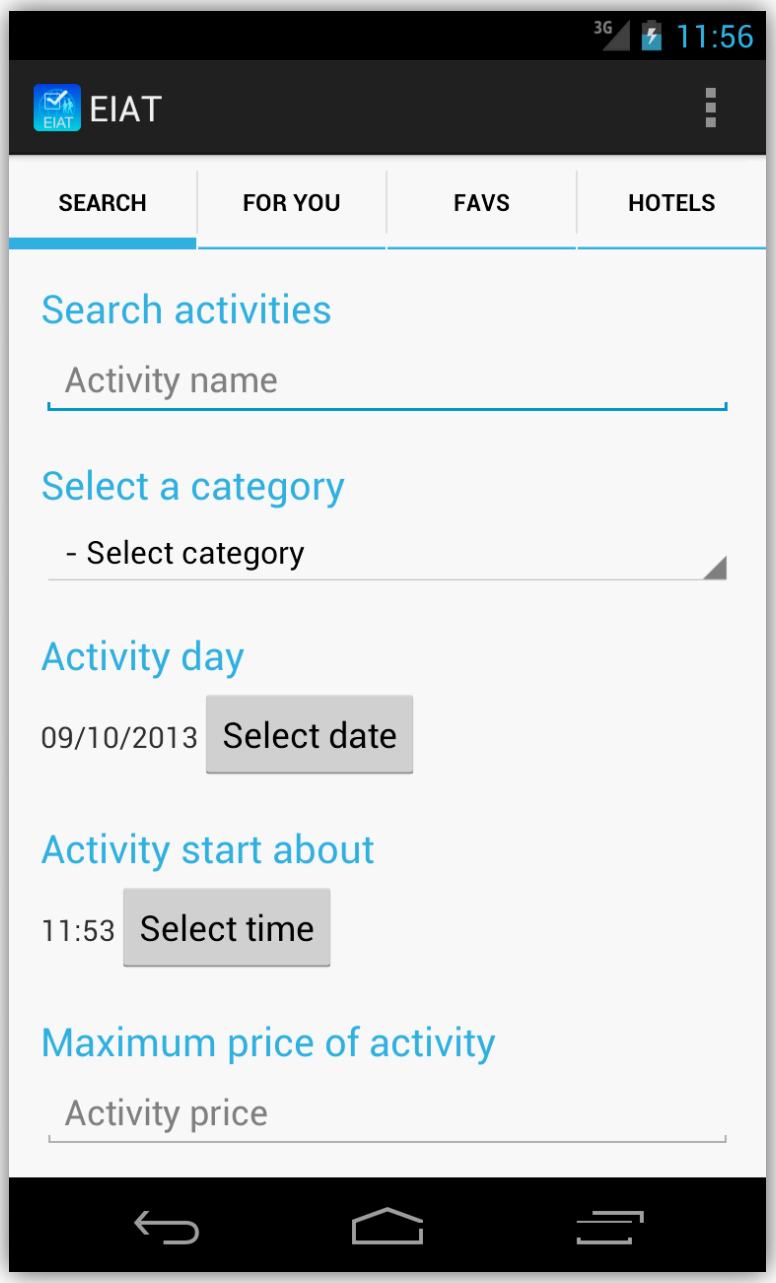


Imagen 13.81. Android. Pestaña search

Existe un menú contextual con el que es posible cerrar la sesión de usuario y acceder al *Acerca de* de la aplicación.

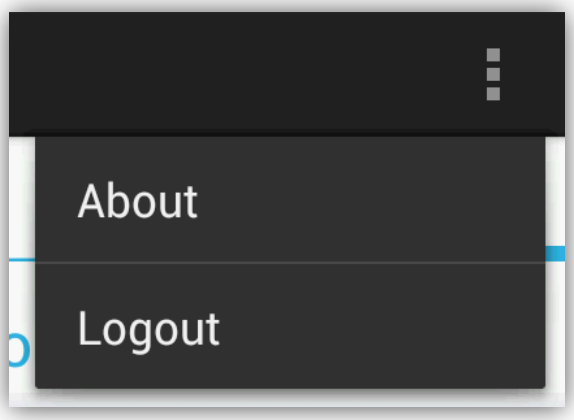


Imagen 13.82. Android. Menú contextual

Para buscar una actividad debemos posicionarnos en la pestaña *Search*. A continuación debemos rellenar al menos uno de los campos, el nombre de la actividad, la categoría, el día y la hora a realizar la actividad y el precio máximo de la actividad. Para seleccionar el día y la hora pulsaremos sobre los botones correspondientes y nos aparecerán unos diálogos para completar los datos gráficamente.

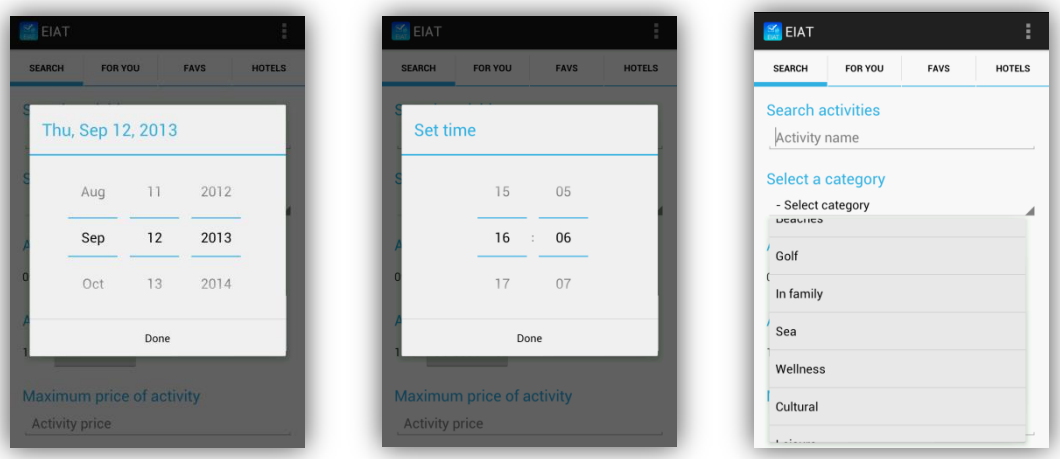


Imagen 13.83. Android. Cuadros de diálogo

Cuando hayamos completado los campos debemos pulsar sobre el botón *Search* de la parte inferior para comenzar el proceso de búsqueda.

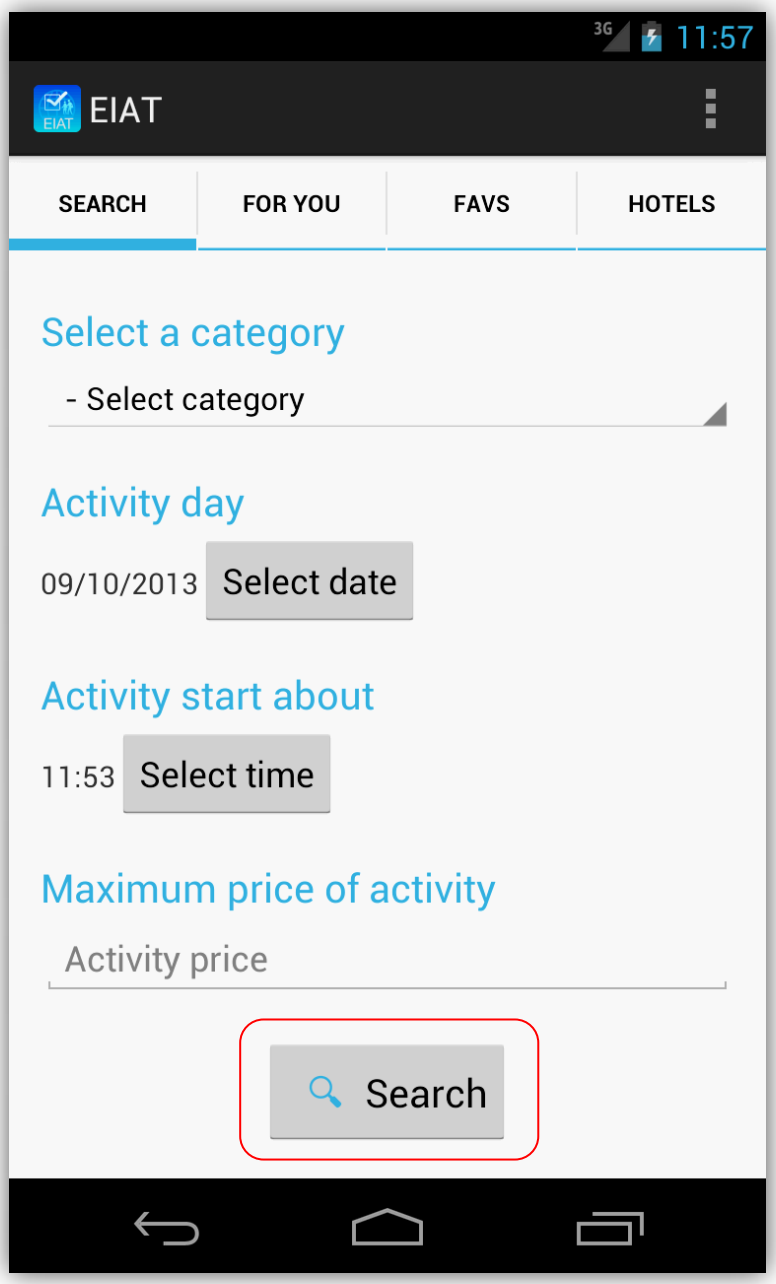


Imagen 13.84. Android. Pestaña search. Comienzo

Pestaña For you

En la pestaña *For you* nos encontramos con dos opciones, la primera obtener una recomendación personalizada y la segunda ver las *Top 10 activities*.

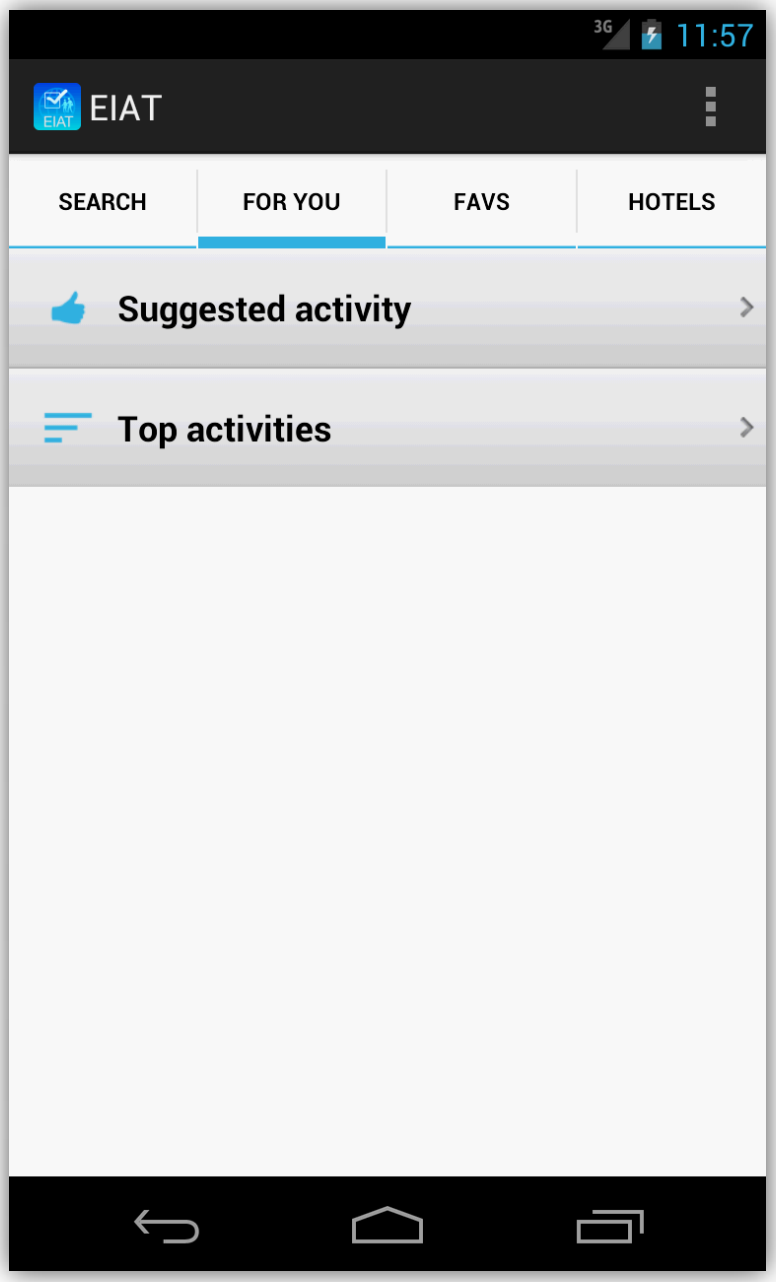


Imagen 13.85. Android. Pestaña For you

Para obtener una recomendación personalizada sólo debemos hacer pulsar sobre el elemento de la lista y se nos mostrará una actividad adecuada a nuestros gustos.

Si deseamos ver las Top 10 activities debemos pulsar sobre ese ítem y veremos la lista de actividades más populares del sistema.

Imagen 13.86. Android. Pestaña For you. Top ten activities

Para visualizar el detalle de cualquiera de las actividades sólo debemos pulsar sobre ella. Si por el contrario deseamos volver al menú anterior deberemos pulsar sobre el botón *Back*.

Pestaña Favs

En la pestaña *Favs* se muestran las actividades que anteriormente hemos añadido a nuestra lista de favoritos.

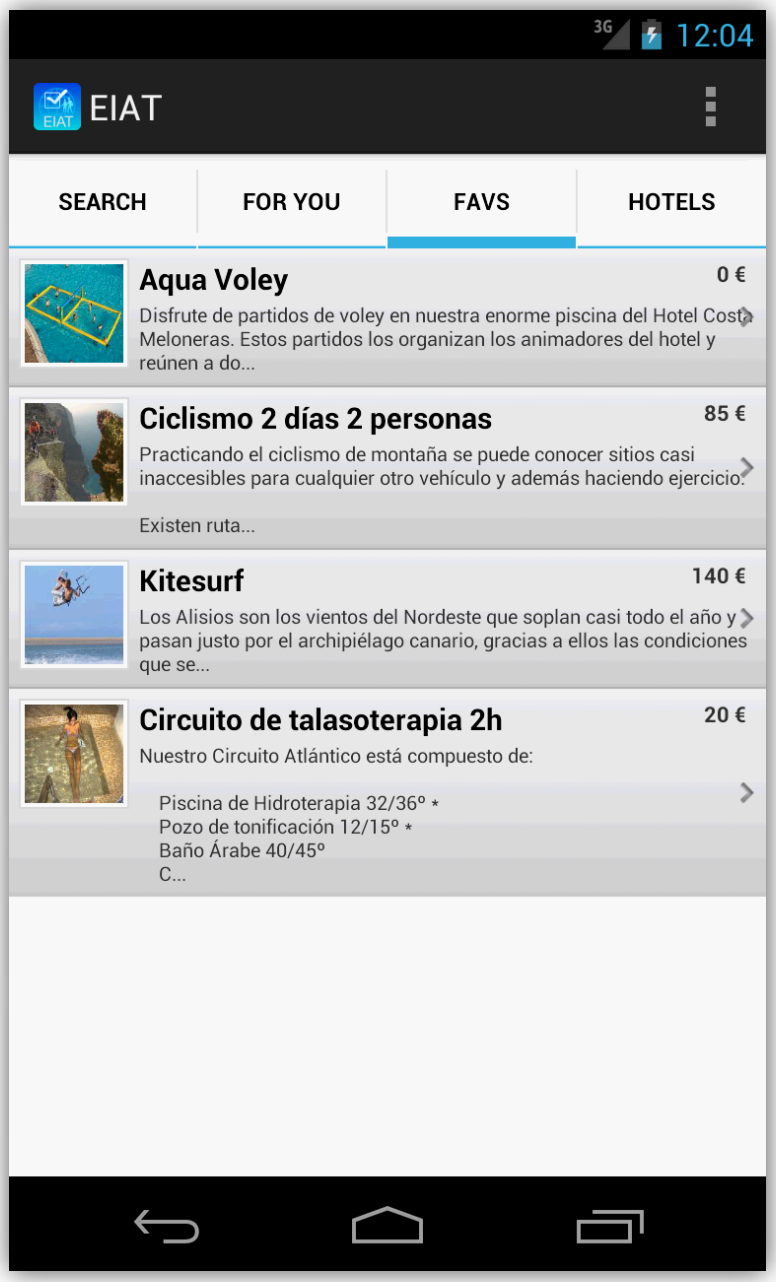


Imagen 13.87. Android. Pestaña Favs

Pestaña Hotels

La pestaña *Hotels* tiene dos comportamientos en función de si se está unido a un hotel o no.

En caso de que no estemos unidos a un hotel se nos mostrará la siguiente interfaz de búsqueda. Para unirnos a un determinado hotel debemos introducir su nombre o parte de su nombre y pulsar sobre *Search*.

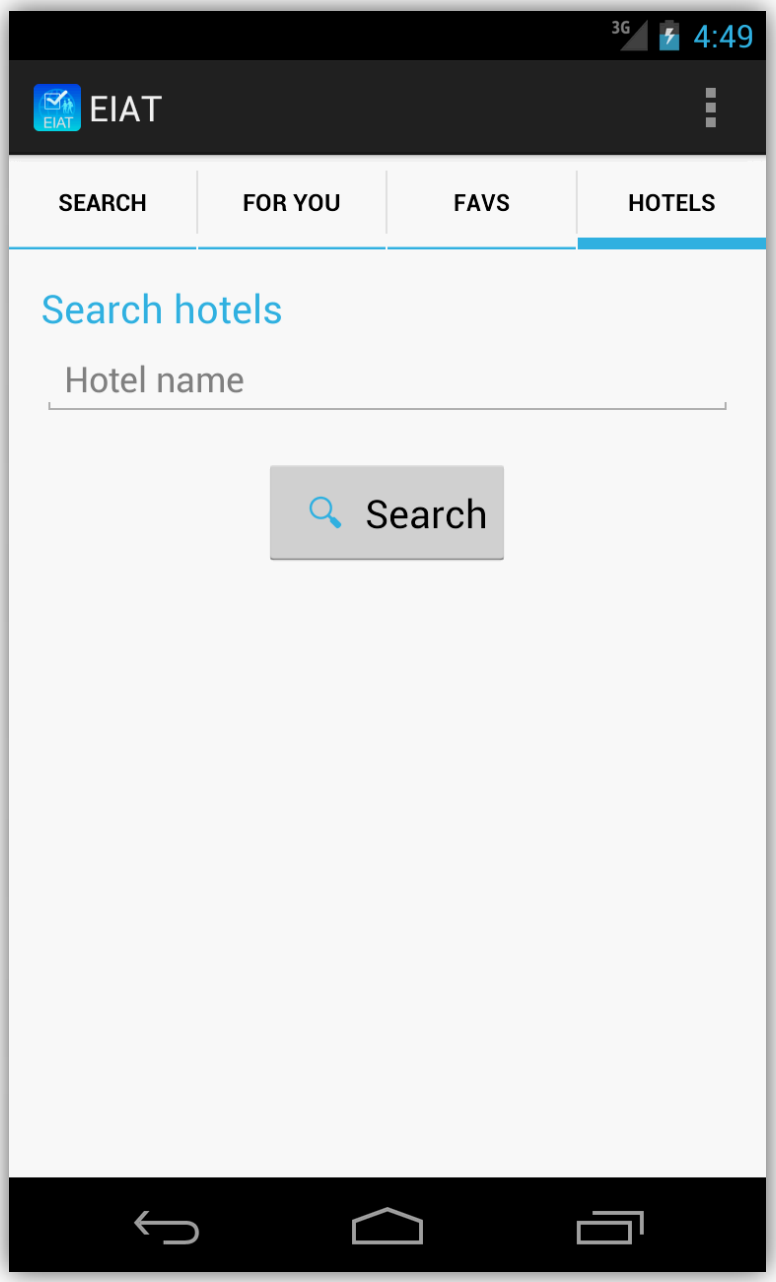


Imagen 13.88. Android. Pestaña hotels. Búsqueda de hotel

Cuando finalice la búsqueda se nos mostrará una lista con los hoteles que coinciden con el nombre proporcionado.

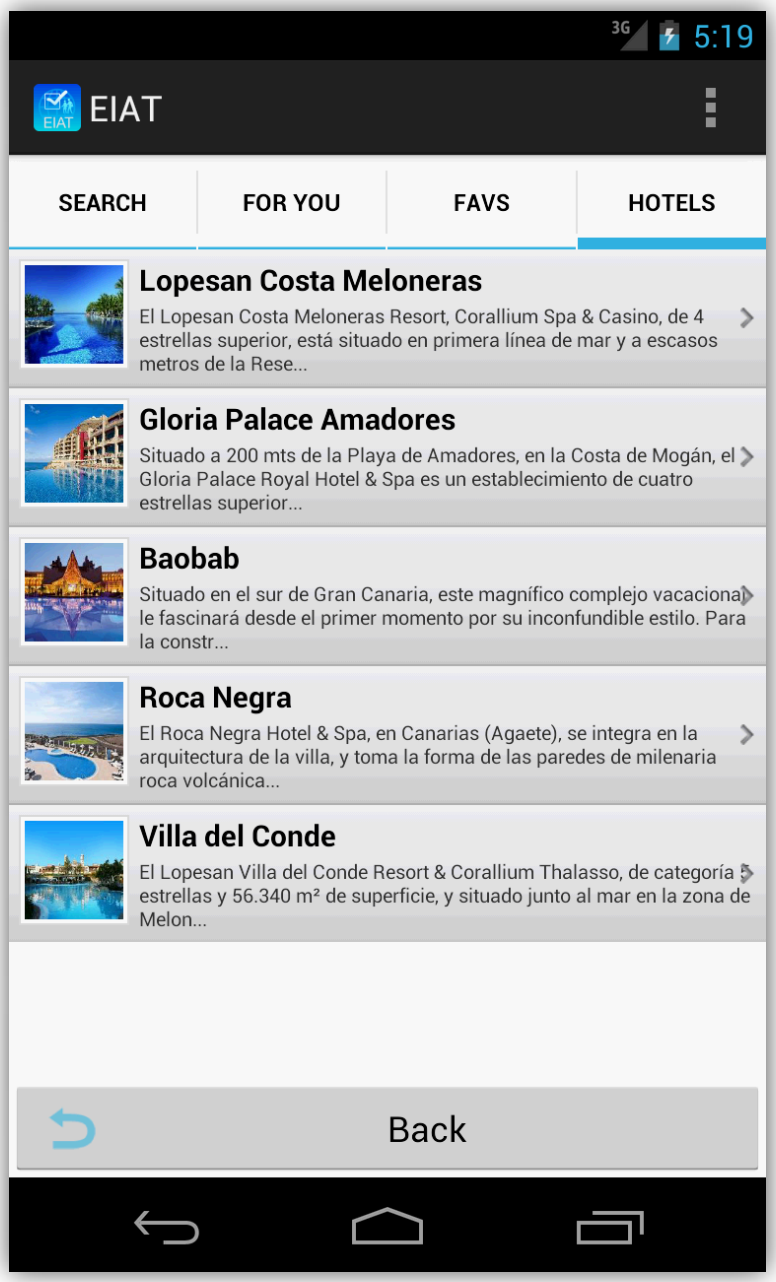


Imagen 13.89. Android. Pestaña Hotels. Lista de hoteles

Para unirnos a uno de ellos sólo debemos pulsar sobre él. Si todo va bien se nos mostrará el siguiente mensaje:

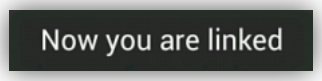


Imagen 13.90. Android. Mensaje de unión a hotel

Por el contrario, si resulta que sí nos encontramos unidos a un hotel se nos mostrará una interfaz similar a esta:

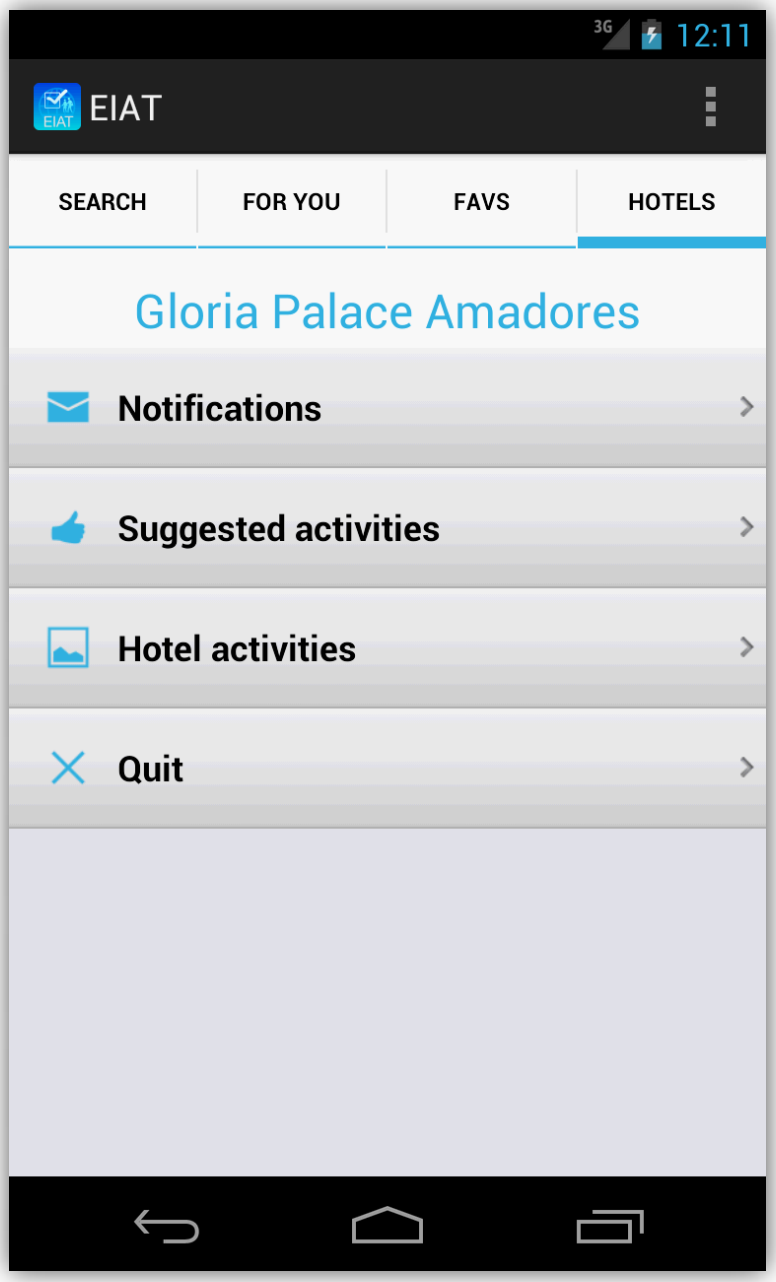


Imagen 13.91. Android. Pestaña Hotels. Menú de hotel

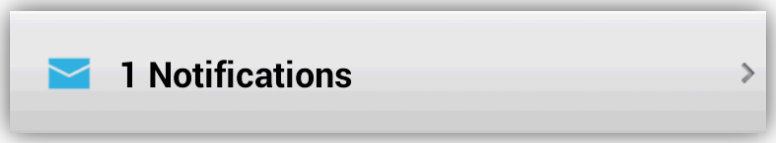


Imagen 13.92. Android. Pestaña hotels. Notificación

Desde ella podemos comprobar si tenemos y acceder a las notificaciones del hotel, ver las actividades, las actividades sugeridas y desunirnos.

Visualizar notificaciones

Para visualizarlas debemos pulsar sobre *Notifications* de la pestaña *Hotels*. Tras esto, se nos mostrará la lista de notificaciones del hotel con el siguiente formato:

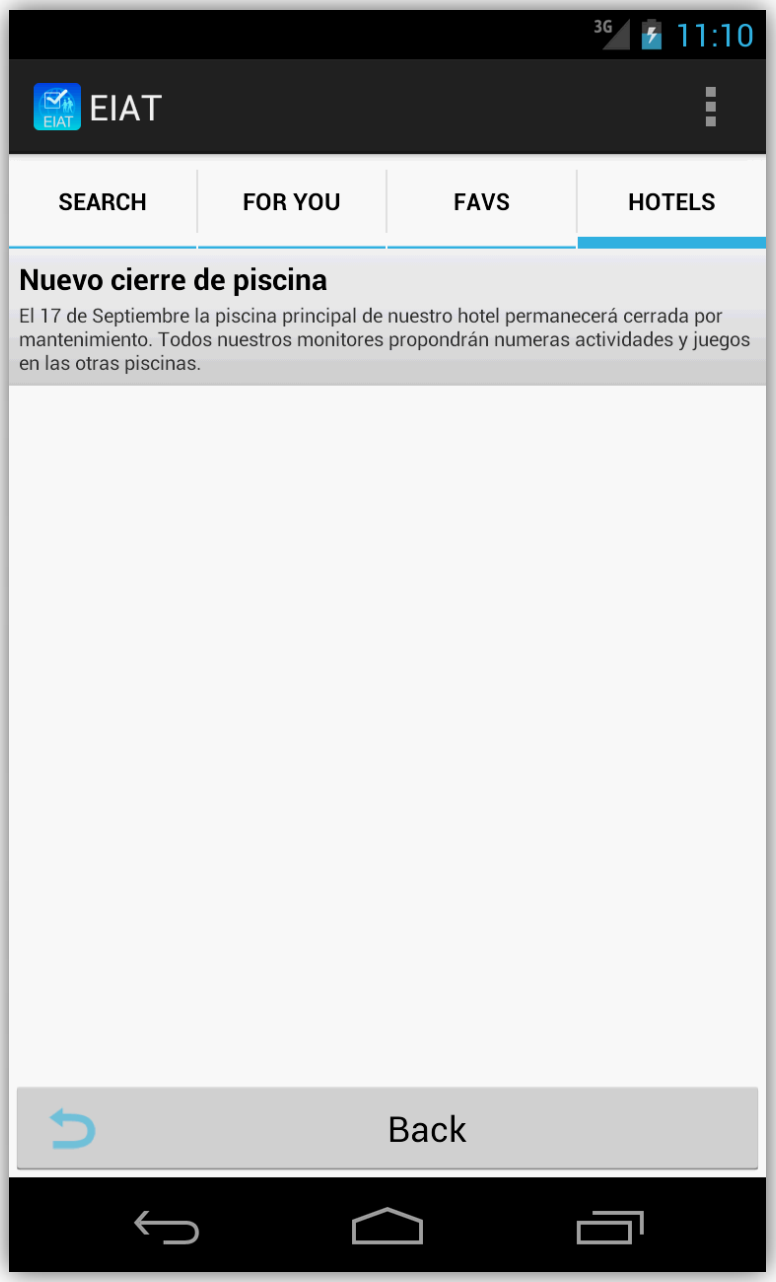


Imagen 13.93. Android. Pestaña hotels. Lista de notificaciones

Desunirnos del hotel

Si deseamos desunirnos del hotel debemos pulsar sobre *Quit* del menú de opciones de la pestaña *Hotels*.

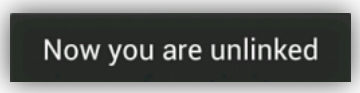


Imagen 13.94. Android. Mensaje de desunión del hotel

Detalle de una actividad

En el detalle de una actividad se nos muestran todos los datos de la misma: nombre, descripción, empresa a la que pertenece, fechas, días y horas a las que está activa, precio y teléfono. Además es posible localizarlas mediante Google Maps, añadirlas o quitarlas de favoritos y votarlas.

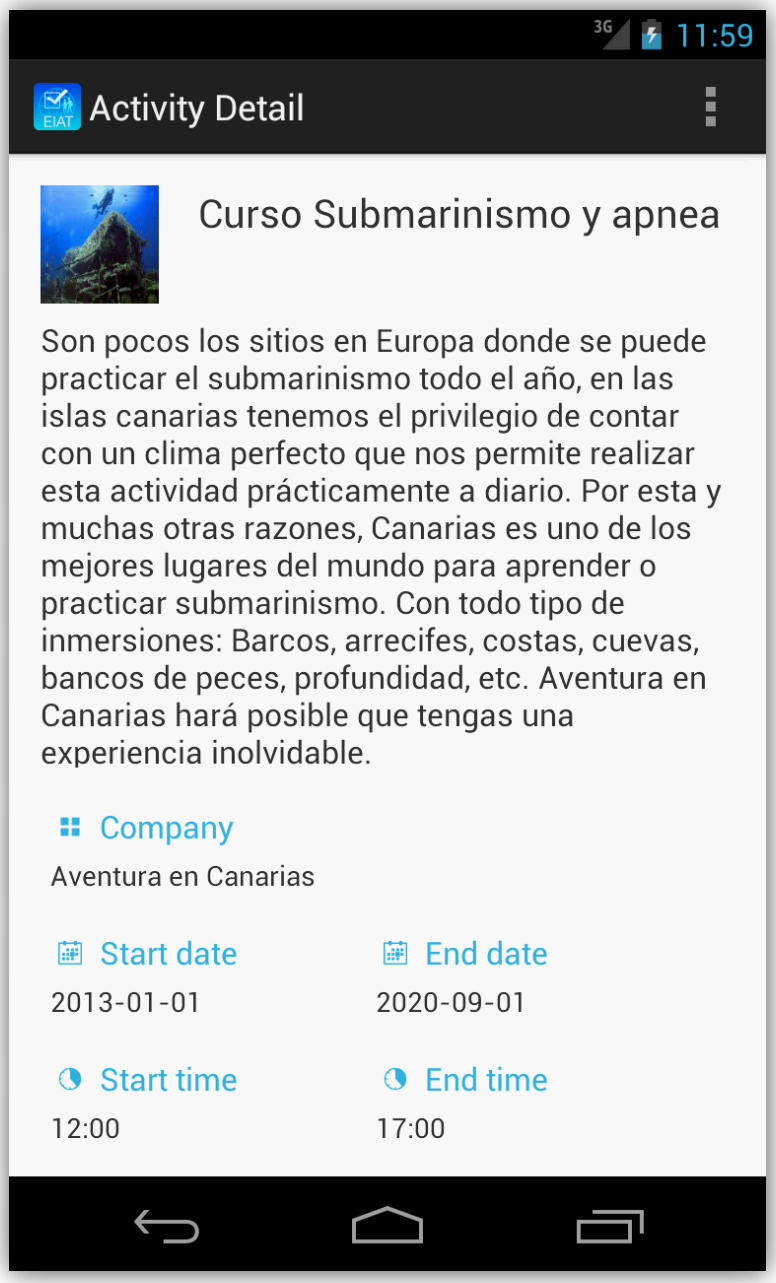


Imagen 13.95. Android. Detalle de actividad (arriba)

Si nos desplazamos hacia abajo de la actividad podremos encontrarnos con el resto de la información y los botones de interacción.

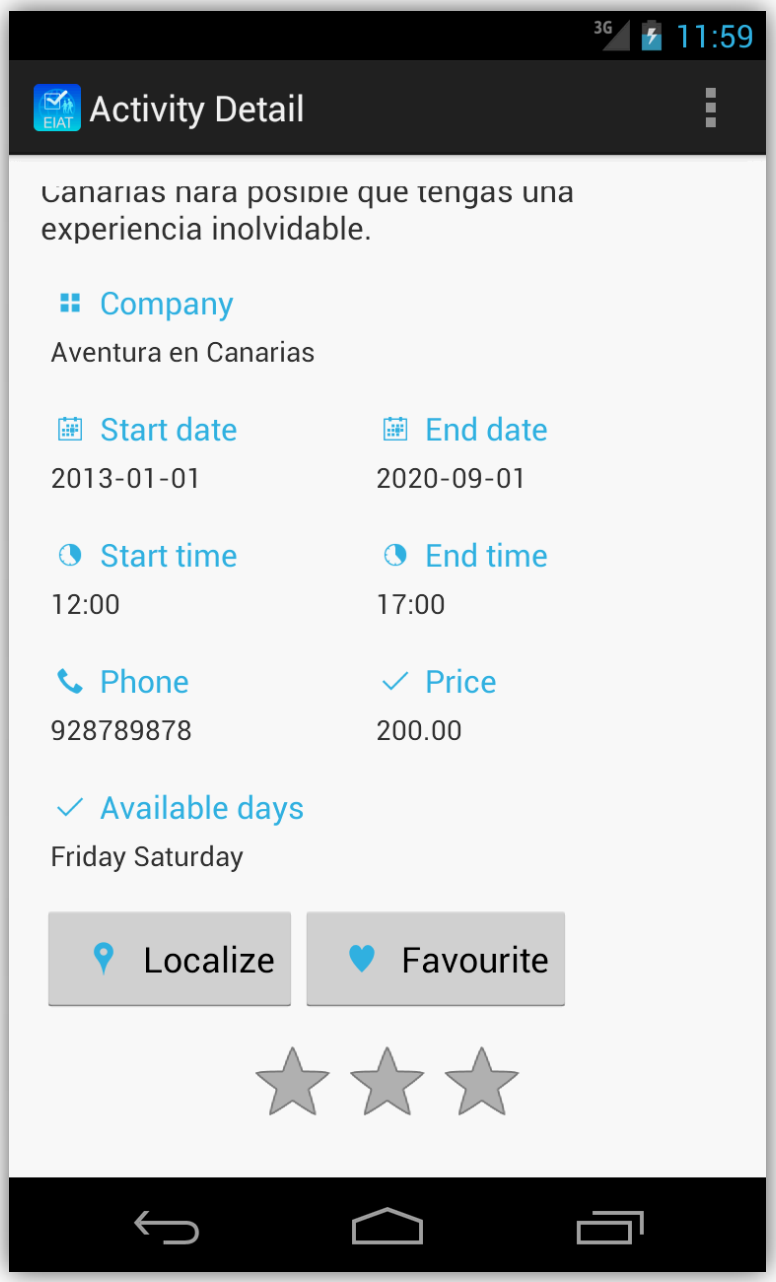


Imagen 13.96. Android. Detalle de actividad (abajo)

En caso de que deseemos eliminar la actividad de nuestra lista de favoritos debemos pulsar sobre *Unfavourite*.

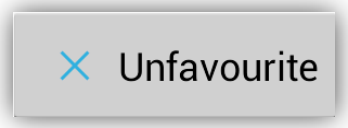


Imagen 13.97. Android. Detalle de actividad. Botón quitar de favoritos

Para votar la actividad debemos pulsar sobre el número de estrellas que deseemos otorgarle.

Imagen 13.98. Android. Detalle de actividad. Votación

Localización de una actividad

Para localizar una actividad debemos pulsar sobre el botón *Localize* en el detalle de la actividad. Con esta acción se nos mostrará un mapa indicándonos donde se realiza.

Imagen 13.99. Android. Localización de actividad